32-Bit RISC Microprocessor TX19 Family

# TMP1962F10AXBG

## 1. Features

The TX19 is a family of high-performance 32-bit microprocessors that offers the speed of a 32-bit RISC solution with the added advantage of a significantly reduced code size of a 16-bit architecture. The instruction set of the TX19 includes as a subset the 32-bit instructions of the TX39, which is based on the MIPS R3000A™ architecture. Additionally, the TX19 supports the MIPS16 Application-Specific Extensions (ASE) for improved code density.

The TMP1962 is built on a TX19 core processor and a selection of intelligent peripherals. The TMP1962 is suitable for low-voltage and low-power applications.

Features of the TMP1962 include the following:

(1) TX19 core processor

1) Two instruction set architecture (ISA) modes: 16-bit ISA for code density and 32-bit ISA for speed
   - The 16-bit ISA is object-code compatible with the code-efficient MIPS16 ASE.
   - The 32-bit ISA is object-code compatible with the high-performance TX39 family.

2) High performance combined with low power consumption

— High performance
   - Single clock cycle execution for most instructions
   - 3-operand computational instructions for high instruction throughput
   - 5-stage pipeline
   - On-chip high-speed memory
   - DSP function: Executes 32-bit x 32-bit multiplier operations in a single clock cycle.

— Low power consumption
- Optimized design using a low-power cell library
- Programmable standby modes in which processor clocks are stopped

3) Fast interrupt response suitable for real-time control
- Distinct starting locations for each interrupt service routine
- Automatically generated vectors for each interrupt source
- Automatic updates of the interrupt mask level

(2) On-chip ROM/RAM

| Product | On-chip ROM | On-chip RAM |
|---|---|---|
| TMP1962C10BXBG | 1 Mbyte | 40 kbyte |
| TMP1962F10AXBG | 1 Mbyte (Flash) | 40 Kbyte |

Collection function on ROM (8 words x 8 blocks)

(3) External memory expansion
- 16-Mbyte off-chip address space for code and data
- External bus interface with dynamic bus sizing for 8-bit and 16-bit data ports (Separate bus/multiplex bus)

(4) 8-channel DMA controller
- Interrupt- or software-triggered
- DMA transfers between on-chip or external memory and I/O module

(5) 12-channel 8-bit timer
- 8-bit/16-bit/24-bit/32-bit interval timer mode
- 8-bit PWM mode
- 8-bit PPG mode

(6) 4-channel 16-bit timer
- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit PPG output
- Input capture function
- 2-channel dual input counter function

(7) 32-bit input capture
- 8-channel 32-bit input capture register
- 8-channel 32-bit compare register
- 1-channel 32-bit time base timer

(8) 7-channel general-purpose serial interface
Either UART mode or synchronous transfer mode can be selected.

(9) 1-channel serial bus interface
Either $I^2C$ bus mode or clock-synchronous mode can be selected.

(10) 24-channel 10-bit A/D converter (with internal sample/hold)

- External trigger start function
- Fixed channel/scan mode
- Single/repeat mode
- Timer monitor function

(11) Watchdog timer

(12) 4-channel chip select/wait controller

(13) Interrupt sources

- 4 CPU interrupts: software interrupt instruction
- 55 internal interrupts: 7 priority levels, with the exception of the watchdog timer interrupt
- 25 external interrupts: 7 priority levels, with the exception of the NMI interrupt
  1 used for an interrupt source and 14 used for KWUP

(14) 202-pin input/output ports

(15) Four standby modes

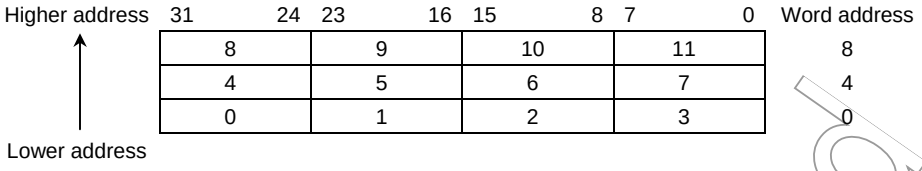
- IDLE (HALT, DOZE), STOP

(16) Clock generator

- On-chip PLL (x3)
- Clock gear: Divides the operating speed of the CPU by 1/2, 1/4 or 1/8

(17) Endian ..... Bi-Endian

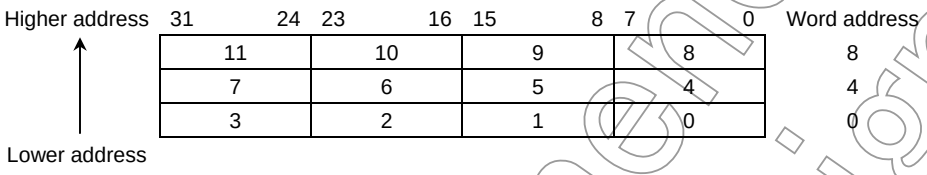
Big-endian




Byte 0 is the most significant byte (MSB) (bits 31-24)

The address of a word data item is the address of its MSB (byte 0).

Little-endian




Byte 0 is the least significant byte (LSB) (bits 7-0)

The address of a word data item is the address of its LSB (byte 0).

(18) Operating frequency

40.5 MHz (Vcc = 2.2 V to 2.7 V)

(19) Package

P-FBGA281 (13 x 13 x 0.65 mm pitch)

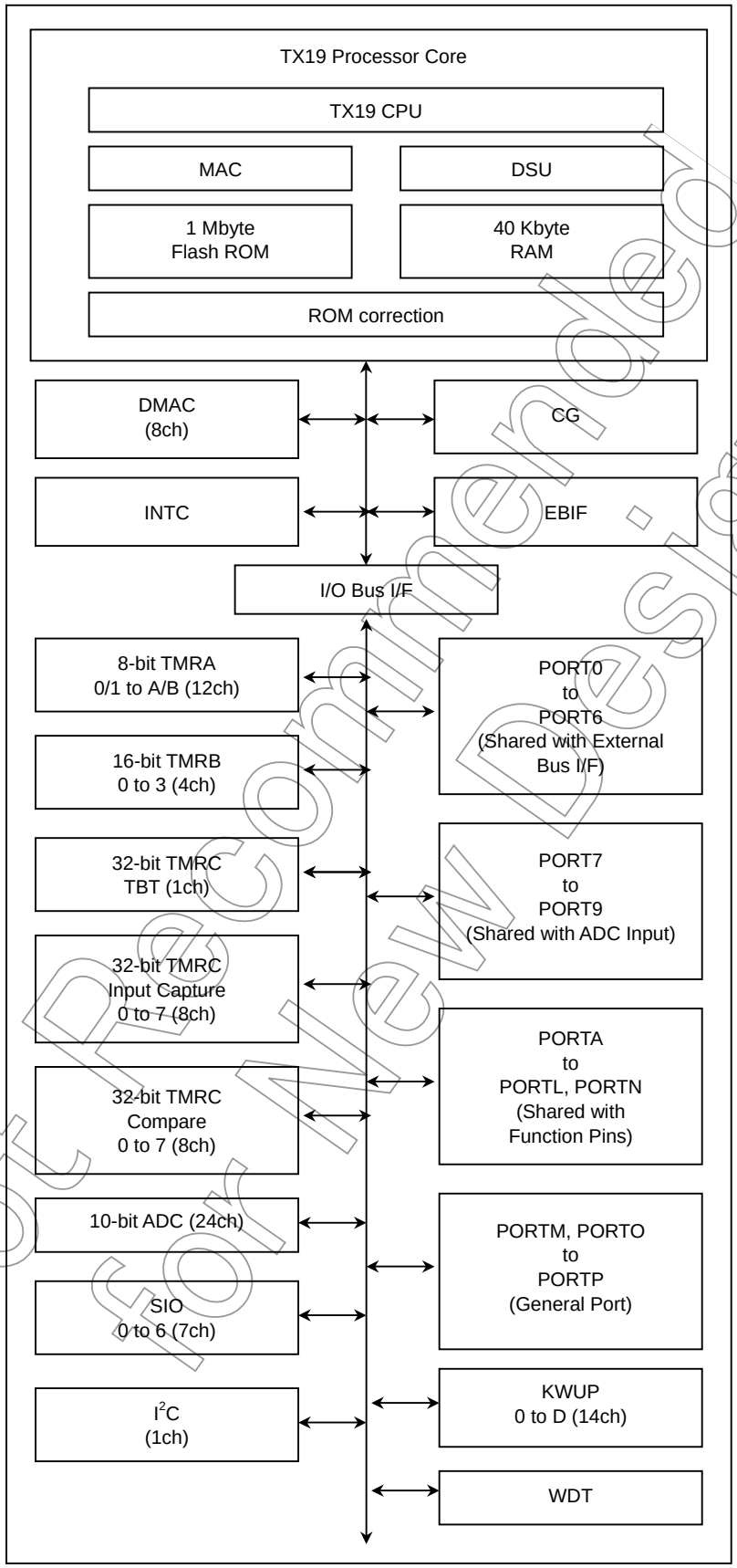



Figure 1.1 TMP1962F10AXBG Block Diagram

# 2. Pin Assignment

This section contains pin assignments for the TMP1962F10AXBG as well as brief description of the TMP1962F10AXBG input and output signals.

## 2.1 Pin Assignment

The following illustrates the TMP1962F10AXBG pin assignment.

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | A15 | A16 | A17 | |
| B1 | B2 | B3 | B4 | B5 | B6 | B7 | B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 | B16 | B17 | B18 |
| C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 | C12 | C13 | C14 | C15 | C16 | C17 | C18 |
| D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10 | D11 | D12 | D13 | D14 | D15 | D16 | D17 | D18 |
| E1 | E2 | E3 | E4 | E5 | E6 | E7 | E8 | E9 | E10 | E11 | E12 | E13 | E14 | E15 | E16 | E17 | E18 |
| F1 | F2 | F3 | F4 | F5 | | F7 | F8 | F9 | F10 | F11 | F12 | | F14 | F15 | F16 | F17 | F18 |
| G1 | G2 | G3 | G4 | G5 | G6 | | | | | | | G13 | G14 | G15 | G16 | G17 | G18 |
| H1 | H2 | H3 | H4 | H5 | H6 | | | | | | | H13 | H14 | H15 | H16 | H17 | H18 |
| J1 | J2 | J3 | J4 | J5 | J6 | | | | | | | J13 | J14 | J15 | J16 | J17 | J18 |
| K1 | K2 | K3 | K4 | K5 | K6 | | | | | | | K13 | K14 | K15 | K16 | K17 | K18 |
| L1 | L2 | L3 | L4 | L5 | L6 | | | | | | | L13 | L14 | L15 | L16 | L17 | L18 |
| M1 | M2 | M3 | M4 | M5 | M6 | | | | | | | M13 | M14 | M15 | M16 | M17 | M18 |
| N1 | N2 | N3 | N4 | N5 | | N7 | N8 | N9 | N10 | N11 | N12 | | N14 | N15 | N16 | N17 | N18 |
| P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 | P11 | P12 | P13 | P14 | P15 | P16 | P17 | P18 |
| R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 | R9 | R10 | R11 | R12 | R13 | R14 | R15 | R16 | R17 | R18 |
| T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 | T11 | T12 | T13 | T14 | T15 | T16 | T17 | T18 |
| U1 | U2 | U3 | U4 | U5 | U6 | U7 | U8 | U9 | U10 | U11 | U12 | U13 | U14 | U15 | U16 | U17 | U18 |
| | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | V10 | V11 | V12 | V13 | V14 | V15 | V16 | V17 | |

Figure 2.1 Pin Assignment (P-FBGA281)

The following provides a pin cross reference by pin number.

Table 2.1 Pin Cross Reference by Pin Number (1/2)

| Pin No. | Pin Name | Pin No. | Pin Name | Pin No. | Pin Name | Pin No. | Pin Name | Pin No. | Pin Name |
|---|---|---|---|---|---|---|---|---|---|
| A1 | NC | A13 | PK1/KEY1 | B8 | P75/AIN5 | C2 | PCST3 (DSU) | C14 | PK6/KEY6 |
| A2 | VREFL | A14 | PI1/INT1 | B9 | PL0/TA4IN | C3 | P92/AIN18 | C15 | PI5/INT9 |
| A3 | P90/AIN16 | A15 | PI3/INT3 | B10 | PL3/TAAIN | C4 | P95/AIN21 | C16 | TCK (JTAG) |
| A4 | P93/AIN19 | A16 | PI6/INTA | B11 | PM1 | C5 | P82/AIN10 | C17 | CVCC2 |
| A5 | P80/AIN8 | A17 | X2 | B12 | PM4 | C6 | P85/AIN13 | C18 | XT2 |
| A6 | P83/AIN11 | B1 | AVCC31 | B13 | PK2/KEY2 | C7 | P72/AIN2 | D1 | SDAO/TPC (DSU) |
| A7 | P70/AIN0 | B2 | VREFH | B14 | PI2/INT2 | C8 | AVSS | D2 | PCST2 (DSU) |
| A8 | P74/AIN4 | B3 | P91/AIN17 | B15 | PI4/INT4 | C9 | PL1/TA6IN | D3 | SDI/$\overline{\text{DINT}}$ (DSU) |
| A9 | NC | B4 | P94/AIN20 | B16 | PI7 | C10 | PL4/TB0IN0 | D4 | DVCC2 |
| A10 | PL2/TA8IN | B5 | P81/AIN9 | B17 | CVSS | C11 | PM2 | D5 | P96/AIN22 |
| A11 | PM0 | B6 | P84/AIN12 | B18 | X1 | C12 | PM5 | D6 | P86/AIN14 |
| A12 | PK0/KEY0 | B7 | P71/AIN1 | C1 | PCST0 (DSU) | C13 | PK3/KEY3 | D7 | P73/AIN3 |

Table 2.1 Pin Cross Reference by Pin Number (2/2)

| Pin No. | Pin Name | Pin No. | Pin Name | Pin No. | Pin Name | Pin No. | Pin Name | Pin No. | Pin Name |
|---|---|---|---|---|---|---|---|---|---|
| D8 | DVCC2 | F18 | P44/SCOUT | K14 | P12/D10/AD10 | N18 | DVSS | T8 | PD4/TXD4 |
| D9 | DVSS | G1 | $\overline{\text{RESET}}$ | K15 | P13/D11/AD11 | P1 | PP0 | T9 | PC0/TXD0 |
| D10 | PL5/TB0IN1 | G2 | TEST5 | K16 | P14/D12/AD12 | P2 | PB2/TB2IN0/INT5 | T10 | PC3/TXD1 |
| D11 | PM3 | G3 | FVCC2 | K17 | DVCC33 | P3 | PB3/TB2IN1/INT6 | T11 | PH4/TCOUT4 |
| D12 | PM6 | G4 | FVSS | K18 | P15/D13/AD13 | P4 | PB4/TB2OUT | T12 | PE2/SCLK5/ $\overline{\text{CTS5}}$ |
| D13 | PK4/KEY4 | G5 | PJ0/INT0 | L1 | FVCC3 | P5 | PB5/TB3IN0/INT7 | T13 | PE5/KEYB |
| D14 | PK7/KEY7 | G6 | BW0 | L2 | PO1 | P6 | PG5/TC5IN | T14 | P53/A3 |
| D15 | DVCC34 | G13 | $\overline{\text{TRST}}$ | L3 | PO2 | P7 | PG7/TC7IN | T15 | P56/A6 |
| D16 | TDI (JTAG) | G14 | CAP1 | L4 | PO3 | P8 | PD6/SCLK4/ $\overline{\text{CTS4}}$ | T16 | P62/A10 |
| D17 | TDO (JTAG) | G15 | P41/ $\overline{\text{CS1}}$ | L5 | PO4 | P9 | PC2/SCLK0/ $\overline{\text{CTS0}}$ | T17 | P65/A13 |
| D18 | XT1 | G16 | P37/ALE | L6 | PO7 | P10 | PC5/SCLK1/ $\overline{\text{CTS1}}$ | T18 | P20/A16/A0 |
| E1 | DCLK (DSU) | G17 | P35/ $\overline{\text{BUSAK}}$ | L13 | TEST3 | P11 | PH6/TCOUT6 | U1 | PA0/TA0IN |
| E2 | PCST1 (DSU) | G18 | FVCC2 | L14 | P06/D6/AD6 | P12 | NC | U2 | PA3/TA3OUT |
| E3 | $\overline{\text{DBGE}}$ | H1 | $\overline{\text{NMI}}$ | L15 | FVCC2 | P13 | P50/A0 | U3 | PA6/TA9OUT |
| E4 | PJ3/INTLV | H2 | DVCC31 | L16 | P07/D7/AD7 | P14 | P51/A1 | U4 | PF1/SI/SCL |
| E5 | PJ4/ENDIAN | H3 | PN7 | L17 | P10/D8/AD8 | P15 | P54/A4 | U5 | PF5/ $\overline{\text{DREQ3}}$ |
| E6 | P97/AIN23 | H4 | BW1 | L18 | P11/D9/AD9 | P16 | P23/A19/A3 | U6 | PG2/TC2IN |
| E7 | P87/AIN15 | H5 | $\overline{\text{PLLOFF}}$ | M1 | PO0 | P17 | P24/A20/A4 | U7 | PD2/RXD3 |
| E8 | P76/AIN6 | H6 | TEST1 | M2 | PP5 | P18 | P25/A21/A5 | U8 | DVCC32 |
| E9 | P77/AIN7 | H13 | TEST2 | M3 | PP6 | R1 | PB0/TB0OUT | U9 | PC7/RXD2 |
| E10 | PL6/TB1IN0 | H14 | P31/ $\overline{\text{WR}}$ | M4 | PP7 | R2 | PB1/TB1OUT | U10 | PH1/TCOUT1 |
| E11 | PL7/TB1IN1 | H15 | P32/ $\overline{\text{HWR}}$ | M5 | PB7/TB3OUT | R3 | PE3/ $\overline{\text{DREQ2}}$ | U11 | PH3/TCOUT3 |
| E12 | PM7 | H16 | P33/WAIT/RDY | M6 | DVCC32 | R4 | PF4/ $\overline{\text{DACK2}}$ | U12 | PE1/RXD5 |
| E13 | PK5/KEY5 | H17 | P30/ $\overline{\text{RD}}$ | M13 | TEST4 | R5 | PF7/TBTIN | U13 | PE4/KEYA |
| E14 | NC | H18 | P40/ $\overline{\text{CS0}}$ | M14 | P02/D2/AD2 | R6 | PG4/TC4IN | U14 | DVCC32 |
| E15 | TMS (JTAG) | J1 | PN2/SCLK6/ $\overline{\text{CTS6}}$ | M15 | FVSS | R7 | PG6/TC6IN | U15 | P57/A7 |
| E16 | CVCCH | J2 | PN3 | M16 | P03/D3/AD3 | R8 | PD5/RXD4 | U16 | P63/A11 |
| E17 | NC | J3 | PN4 | M17 | P04/D4/AD4 | R9 | PC1/RXD0 | U17 | P66/A14 |
| E18 | DVCC2 | J4 | PN5 | M18 | P05/D5/AD5 | R10 | PC4/RXD1 | U18 | DVCC33 |
| F1 | DVSS | J5 | PN6 | N1 | PP1 | R11 | PH5/TCOUT5 | V2 | PA2/TA2IN |
| F2 | $\overline{\text{DRESET}}$ | J6 | DVCC2 | N2 | PP2 | R12 | PH7/TCOUT7 | V3 | PA5/TA7OUT |
| F3 | SYSRDY | J13 | FVSS | N3 | PP3 | R13 | PE6/KEYC | V4 | PF0/SO/SDA |
| F4 | PJ1/BUSMD | J14 | P16/D14/AD14 | N4 | PP4 | R14 | P52/A2 | V5 | PG0/TC0IN |
| F5 | PJ2/ $\overline{\text{BOOT}}$ | J15 | DVSS | N5 | PB6/TB3IN1/INT8 | R15 | P55/A5 | V6 | PG1/TC1IN |
| F7 | AVSS | J16 | P17/D15/AD15 | N7 | DVSS | R16 | P61/A9 | V7 | PD1/TXD3 |
| F8 | AVSS | J17 | P36/ R / $\overline{\text{W}}$ | N8 | PD7/KEY8 | R17 | P21/A17/A1 | V8 | PD0/SCLK2/ $\overline{\text{CTS2}}$ |
| F9 | AVCC32 | J18 | P34/ $\overline{\text{BUSRQ}}$ | N9 | DVCC2 | R18 | P22/A18/A2 | V9 | PC6/TXD2 |
| F10 | DVCC34 | K1 | PN0/TXD6 | N10 | DVSS | T1 | PA1/TA1OUT | V10 | PH0/TCOUT0 |
| F11 | PI0/ $\overline{\text{ADTRG}}$ | K2 | PN1/RXD6 | N11 | RSTPUP | T2 | PA4/TA5OUT | V11 | PH2/TCOUT2 |
| F12 | DVSS | K3 | PO5 | N12 | DVSS | T3 | PA7/TABOUT | V12 | PE0/TXD5 |
| F14 | CAP2 | K4 | PO6 | N14 | P26/A22/A6 | T4 | PF2/SCK | V13 | PE3/KEY9 |
| F15 | P42/ $\overline{\text{CS2}}$ | K5 | FVSS | N15 | P27/A23/A7 | T5 | PF6/ $\overline{\text{DACK3}}$ | V14 | PE7/KEYD |
| F16 | P43/ $\overline{\text{CS3}}$ | K6 | DVSS | N16 | P00/D0/AD0 | T6 | PG3/TC3IN | V15 | P60/A8 |
| F17 | DVCC33 | K13 | TEST0 | N17 | P01/D1/AD1 | T7 | PD3/SCLK3/ $\overline{\text{CTS3}}$ | V16 | P64/A12 |
| | | | | | | | | V17 | P67/A15 |

## 2.2 Pin Usage Information

Table 2.2 lists input and output pins of the TMP1962F10AXBG.

Table 2.2 Pin Names and Function (1/6)

| Pin Name | # of Pins | Type | Function |
|---|---|---|---|
| P00-P07 | 8 | Input/Output | Port 0: Individually programmable input or output |
| D0-D7 | | Input/Output | Data (Lower): Bits 0-7 of the data bus (separate bus mode) |
| AD0-D7 | | Input/Output | Address /Data (Lower): Bits 0-7 of the address/data bus (multiplex bus mode) |
| P10-P17 | 8 | Input/Output | Port 1: Individually programmable input or output |
| D8-D15 | | Input/Output | Data (Upper): Bits 8-15 of the data bus (separate bus mode) |
| AD8-AD15 | | Input/Output | Address /Data (Upper): Bits 8-15 of the address/data bus (multiplex bus mode) |
| A8-A15 | | Output | Address: Bits 8-15 of the address bus (multiplex bus mode) |
| P20-P27 | 8 | Input/Output | Port 2: Individually programmable input or output |
| A16-A23 | | Output | Address: Bit 15-23 of the address bus (separate bus mode) |
| A0-A7 | | Output | Address: Bit 0-7 of the address bus (multiplex bus mode) |
| A16-A23 | | Output | Address: Bit 16-23 of the address bus (multiplex bus mode) |
| P30 | 1 | Output | Port 30: Output-only |
| $\overline{RD}$ | | Output | Read Strobe: Asserted during a read operation from an external memory device |
| P31 | 1 | Output | Port 31: Output-only |
| $\overline{WR}$ | | Output | Write Strobe: Asserted during a write operation on D0-D7 |
| P32 | 1 | Input/Output | Port 32: Programmable as input or output (with internal pull-up register) |
| $\overline{HWR}$ | | Output | Higher Write Strobe: Asserted during a write operation on D8-D15 |
| P33 | 1 | Input/Output | Port 33: Programmable as input or output (with internal pull-up resister) |
| $\overline{WAIT}$ | | Input | Wait: Causes the CPU to suspend external bus activity |
| $\overline{RDY}$ | | Input | Ready: Informs the CPU of bus ready condition |
| P34 | 1 | Input/Output | Port 34: Programmable as input or output (with internal pull-up resister) |
| $\overline{BUSRQ}$ | | Input | Bus Request: Asserted by an external bus master to request bus mastership |
| P35 | 1 | Input/Output | Port 35: Programmable as input or output (with internal pull-up resister) |
| $\overline{BUSAK}$ | | Output | Bus Acknowledge: Indicates that the CPU has relinquished the bus in response to $\overline{BUSRQ}$ . |
| P36 | 1 | Input/Output | Port 36: Programmable as input or output (with internal pull-up resister) |
| R/$\overline{W}$ | | Output | Read/Write: Indicates the direction of data transfer on the bus: 1 = read or dummy cycle, 0 = write cycle |
| P37 | 1 | Input/Output | Port 37: Programmable as input or output |
| ALE | | Output | Address Latch Enable (This signal is driven out only when external memory is accessed.) |
| P40 | 1 | Input/Output | Port 40: Programmable as input or output (with internal pull-up resister) |
| $\overline{CS0}$ | | Output | Chip Select 0: Asserted low to enable external devices at programmed addresses |
| P41 | 1 | Input/Output | Port 41: Programmable as input or output (with internal pull-up resister) |
| $\overline{CS1}$ | | Output | Chip Select 1: Asserted low to enable external devices at programmed addresses |
| P42 | 1 | Input/Output | Port 42: Programmable as input or output (with internal pull-up resister) |
| $\overline{CS2}$ | | Output | Chip Select 2: Asserted low to enable external devices at programmed addresses |
| P43 | 1 | Input/Output | Port 43: Programmable as input or output (with internal pull-up resister) |
| $\overline{CS3}$ | | Output | Chip Select 3: Asserted low to enable external devices at programmed addresses |
| P44 | 1 | Input/Output | Port 44: Programmable as input or output |
| SCOUT | | Output | System Clock Output: Drives out a clock signal at the frequency equal to or one half of CPU clock (high-speed or low-speed) |
| P50-P57 | 8 | Input/Output | Port 5: Individually programmable as input or output |
| A0-A7 | | Output | Address: Address bus 0-7 (separate bus mode) |
| P60-P67 | 8 | Input/Output | Port 6: Individually programmable as input or output |
| A8-A15 | | Output | Address: Address bus 8-15 (separate bus mode) |
| P70-P77 | 8 | Input | Port 7: Input-only |
| AN0-AN7 | | Input | Analog Input: Input to the on-chip A/D converter |
| P80-P87 | 8 | Input | Port 8: Input-only |
| AN8-AN15 | | Input | Analog Input: Input to the on-chip A/D converter |

Table 2.2 Pin Names and Function (2/6)

| Pin Name | # of Pins | Type | Function |
|---|---|---|---|
| P90-P97<br>AN16-AN23 | 8 | Input<br>Input | Port 9: Input-only<br>Analog Input: Input to the on-chip A/D converter |
| PI0<br>$\overline{\text{ADTRG}}$ | 1 | Input/Output<br>Input | Port I0: Programmable as input or output<br>AD Trigger: Starts an A/D conversion<br>Schmitt trigger input |
| PI1<br>INT1 | 1 | Input/Output<br>Input | Port I1: Programmable as input or output<br>Interrupt Request 1: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive<br>Schmitt trigger input |
| PI2<br>INT2 | 1 | Input/Output<br>Input | Port I2: Programmable as input or output<br>Interrupt Request 2: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive<br>Schmitt trigger input |
| PI3<br>INT3 | 1 | Input/Output<br>Input | Port I3: Programmable as input or output<br>Interrupt Request 3: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive<br>Schmitt trigger input |
| PI4<br>INT4 | 1 | Input/Output<br>Input | Port I4: Programmable as input or output<br>Interrupt Request 4: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive<br>Schmitt trigger input |
| PI5<br>INT9 | 1 | Input/Output<br>Input | Port I5: Programmable as input or output<br>Interrupt Request 9: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive<br>Schmitt trigger input |
| PI6<br>INTA | 1 | Input/Output<br>Input | Port I6: Programmable as input or output<br>Interrupt Request A: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive<br>Schmitt trigger input |
| PI7 | 1 | Input/Output | Port I7: Programmable as input or output |
| PA0<br>TA0IN | 1 | Input/Output<br>Input | Port A0: Programmable as input or output<br>8-Bit Timer 0 Input: Input to 8-bit Timer 0 |
| PA1<br>TA1OUT | 1 | Input/Output<br>Output | Port A1: Programmable as input or output<br>8-Bit Timer 01 Output: Output from either 8-bit Timer 0 or Timer 1 |
| PA2<br>TA2IN | 1 | Input/Output<br>Input | Port A2: Programmable as input or output<br>8-Bit Timer 2 Input: Input to 8-bit Timer 2 |
| PA3<br>TA3OUT | 1 | Input/Output<br>Output | Port A3: Programmable as input or output<br>8-Bit Timer 23 Output: Output from either 8-bit Timer 2 or Timer 3 |
| PA4<br>TA5OUT | 1 | Input/Output<br>Output | Port A4: Programmable as input or output<br>8-Bit Timer 45 Output: Output from either 8-bit Timer 4 or Timer 5 |
| PA5<br>TA7OUT | 1 | Input/Output<br>Output | Port A5: Programmable as input or output<br>8-Bit Timer 67 Output: Output from either 8-bit Timer 6 or Timer 7 |
| PA6<br>TA9OUT | 1 | Input/Output<br>Input | Port A6: Programmable as input or output<br>8-Bit Timer 89 Output: Output from either 8-bit Timer 8 or Timer 9 |
| PA7<br>TABOUT | 1 | Input/Output<br>Output | Port A7: Programmable as input or output<br>8-Bit Timer AB Output: Output from either 8-bit Timer A or Timer B |
| PB0<br>TB0OUT | 1 | Input/Output<br>Output | Port B0: Programmable as input or output<br>16-Bit Timer 0 Output: Output from 16-bit Timer 0 |
| PB1<br>TB1OUT | 1 | Input/Output<br>Output | Port B1: Programmable as input or output<br>16-Bit Timer 1 Output: Output from 16-bit Timer 1 |
| PB2<br>TB2IN0<br>INT5 | 1 | Input/Output<br>Input<br>Input | Port B2: Programmable as input or output<br>16-Bit Timer 2 Input 0: Count/capture trigger input to 16-bit Timer 2<br>Interrupt Request 5: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive |

Table 2.2 Pin Names and Function (3/6)

| Pin Name | # of Pins | Type | Function |
|---|---|---|---|
| PB3<br>TB2IN1<br>INT6 | 1 | Input/Output<br>Input<br>Input | Port B3: Programmable as input or output<br>16-Bit Timer 2 Input 1: Capture trigger input to 16-bit Timer 2<br>Interrupt Request 6: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive |
| PB4<br>TB2OUT | 1 | Input/Output<br>Output | Port B4: Programmable as input or output<br>16-Bit Timer 2 Output: Output from 16-bit Timer 2 |
| PB5<br>TB3IN0<br>INT7 | 1 | Input/Output<br>Input<br>Input | Port B5: Programmable as input or output<br>16-Bit Timer 3 Input 0: Count/capture trigger input to 16-bit Timer 3<br>Interrupt Request 7: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive |
| PB6<br>TB3IN1<br>INT8 | 1 | Input/Output<br>Input<br>Input | Port B6: Programmable as input or output<br>16-Bit Timer 3 Input 1: Capture trigger input to 16-bit Timer 3<br>Interrupt Request 8: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive |
| PB7<br>TB3OUT | 1 | Input/Output<br>Output | Port B7: Programmable as input or output<br>16-Bit Timer 3 Output: Output from 16-bit Timer 3 |
| PC0<br>TXD0 | 1 | Input/Output<br>Output | Port C0: Programmable as input or output<br>Serial Transmit Data 0: Programmable as a push-pull or open-drain output |
| PC1<br>RXD0 | 1 | Input/Output<br>Input | Port C1: Programmable as input or output<br>Serial Receive Data 0 |
| PC2<br>SCLK0<br>$\overline{CTS0}$ | 1 | Input/Output<br>Input<br>Input | Port C2: Programmable as input or output<br>Serial Clock Input/Output 0<br>Serial Clear-to-Send 0<br>Programmable as a push-pull or open-drain output |
| PC3<br>TXD1 | 1 | Input/Output<br>Output | Port C3: Programmable as input or output<br>Serial Transmit Data 1: Programmable as a push-pull or open-drain output |
| PC4<br>RXD1 | 1 | Input/Output<br>Input | Port C4: Programmable as input or output<br>Serial Receive Data 1 |
| PC5<br>SCLK1<br>$\overline{CTS1}$ | 1 | Input/Output<br>Input<br>Input | Port C5: Programmable as input or output<br>Serial Clock Input/Output 1<br>Serial Clear-to-Send 1<br>Programmable as a push-pull or open-drain output |
| PC6<br>TXD2 | 1 | Input/Output<br>Output | Port C6: Programmable as input or output<br>Serial Transmit Data 2: Programmable as a push-pull or open-drain output |
| PC7<br>RXD2 | 1 | Input/Output<br>Input | Port C7: Programmable as input or output<br>Serial Receive Data 2 |
| PD0<br>SCLK2<br>$\overline{CTS2}$ | 1 | Input/Output<br>Input<br>Input | Port D0: Programmable as input or output<br>Serial Clock Input/Output 2<br>Serial Clear-to-Send 2<br>Programmable as a push-pull or open-drain output |
| PD1<br>TXD3 | 1 | Input/Output<br>Output | Port D1: Programmable as input or output<br>Serial Transmit Data 3: Programmable as a push-pull or open-drain output |
| PD2<br>RXD3 | 1 | Input/Output<br>Input | Port D2: Programmable as input or output<br>Serial Receive Data 3 |
| PD3<br>SCLK3<br>$\overline{CTS3}$ | 1 | Input/Output<br>Input<br>Input | Port D3: Programmable as input or output<br>Serial Clock Input/Output 3<br>Serial Clear-to-Send 3<br>Programmable as a push-pull or open-drain output |
| PD4<br>TXD4 | 1 | Input/Output<br>Output | Port D4: Programmable as input or output<br>Serial Transmit Data 4: Programmable as a push-pull or open-drain output |
| PD5<br>RXD4 | 1 | Input/Output<br>Input | Port D5: Programmable as input or output<br>Serial Receive Data 4 |
| PD6<br>SCLK4<br>$\overline{CTS4}$ | 1 | Input/Output<br>Input<br>Input | Port D6: Programmable as input or output<br>Serial Clock Input/Output 4<br>Serial Clear-to-Send 4<br>Programmable as a push-pull or open-drain output |
| PD7<br>KEY8 | 1 | Input/Output<br>Input | Port D7: Programmable as input or output<br>KEY on wake up Input 8: (dynamic pull-up selectable) (with internal pull-up register)<br>Schmitt trigger input |

Table 2.2 Pin Names and Function (4/6)

| Pin Name | # of Pins | Type | Function |
|---|---|---|---|
| PE0<br>TXD5 | 1 | Input/Output<br>Output | Port E0: Programmable as input or output<br>Serial Transmit Data 5: Programmable as a push-pull or open-drain output |
| PE1<br>RXD5 | 1 | Input/Output<br>Input | Port E1: Programmable as input or output<br>Serial Receive Data 5 |
| PE2<br>SCLK5<br>$\overline{CTS5}$ | 1 | Input/Output<br>Input<br>Input | Port E2: Programmable as input or output<br>Serial Clock Input/Output 5<br>Serial Clear-to-Send 5<br>Programmable as a push-pull or open-drain output |
| PE3<br>KEY9 | 1 | Input/Output<br>Input | Port E3: Programmable as input or output<br>KEY on wake up Input 9: (dynamic pull-up selectable) (with internal pull-up register)<br>Schmitt trigger input |
| PE4<br>KEYA | 1 | Input/Output<br>Input | Port E4: Programmable as input or output<br>KEY on wake up Input A: (dynamic pull-up selectable) (with internal pull-up register)<br>Schmitt trigger input |
| PE5<br>KEYB | 1 | Input/Output<br>Input | Port E5: Programmable as input or output<br>KEY on wake up Input B: (dynamic pull-up selectable) (with internal pull-up register)<br>Schmitt trigger input |
| PE6<br>KEYC | 1 | Input/Output<br>Input | Port E6: Programmable as input or output<br>KEY on wake up Input C: (dynamic pull-up selectable) (with internal pull-up register)<br>Schmitt trigger input |
| PE7<br>KEYD | 1 | Input/Output<br>Input | Port C7: Programmable as input or output<br>KEY on wake up Input D: (dynamic pull-up selectable) (with internal pull-up register)<br>Schmitt trigger input |
| PF0<br>SO<br>SDA | 1 | Input/Output<br>Output<br>Input/Output | Port F0: Programmable as input or output<br>Data transmit pin when the Serial Bus Interface is in SIO mode<br>Data transmit/receive pin when the Serial Bus Interface is in I2C mode; programmable as a push-pull or open-drain output<br>Schmitt trigger input |
| PF1<br>SI<br>SCL | 1 | Input/Output<br>Input<br>Input/Output | Port F1: Programmable as input or output<br>Data receive pin when the Serial Bus Interface is in SIO mode<br>Clock input/output pin when the Serial Bus Interface is in I2C mode; programmable as a push-pull or open-drain output<br>Schmitt trigger input |
| PF2<br>SCK | 1 | Input/Output<br>Input/Output | Port F2: Programmable as input or output<br>Clock input/output pin when the Serial Bus Interface is in SIO mode |
| PF3<br>$\overline{DREQ2}$ | 1 | Input/Output<br>Input | Port F3: Programmable as input or output<br>DMA Request 2: DMA transfer request from an external I/O device to DMAC2 |
| PF4<br>$\overline{DACK2}$ | 1 | Input/Output<br>Output | Port F4: Programmable as input or output<br>DMA Acknowledge 2: Acknowledge signal for DMA transfer requested by DREQ2 |
| PF5<br>$\overline{DREQ3}$ | 1 | Input/Output<br>Input | Port F5: Programmable as input or output<br>DMA Request 3: DMA transfer request from an external I/O device to DMAC3 |
| PF6<br>$\overline{DACK3}$ | 1 | Input/Output<br>Output | Port F6: Programmable as input or output<br>DMA Acknowledge 3: Acknowledge signal for DMA transfer requested by DREQ3 |
| PF7<br>TBTIN | 1 | Input/Output<br>Input | Port F7: Programmable as input or output<br>32-bit Time-Base Timer Input: Count input to 32-bit time-base Timer |
| PG0-PG7<br>TC0IN-TC7IN | 8 | Input/Output<br>Input | Port G: Individually programmable as input or output<br>32-Bit Timer Capture Trigger Input |
| PH0-PH7<br>TCOUT0-TCOUT7 | 8 | Input/Output<br>Output | Port H: Individually programmable as input or output<br>32-Bit Timer Compare Match Output |
| PJ0<br>INT0 | 1 | Input/Output<br>Input | Port J0: Programmable as input or output<br>Interrupt Request 0: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive<br>Schmitt trigger input |

Table 2.2 Pin Names and Function (5/6)

| Pin Name | # of Pins | Type | Function |
|---|---|---|---|
| PJ1<br>BUSMD | 1 | Input/Output<br>Input | Port J1: Programmable as input or output<br>External Bus Mode: If this pin is sampled high (DVCC21) at the rising edge of RESET, the TMP1962F10AXBG enters multiplex bus mode. If this pin is sampled low at the rising edge of RESET, the TMP1962F10AXBG enters separate bus mode. During a reset sequence, this pin should be pulled up to a logic 1or pulled down to a logic 0 depending on the bus mode to be used. |
| PJ2<br>$\overline{\text{BOOT}}$ | 1 | Input/Output<br>Input | Port J2: Programmable as input or output<br>Single Boot Mode: If this pin is sampled low at the rising edge of RESET, the TMP1962F10AXBG enters Single Boot mode for re-programming of the on-chip flash. If this pin is sampled high (DVCC21) at the rising edge of RESET, the TMP1962F10AXBG enters NORMAL mode. During a reset sequence, this pin should be pulled up to a logic 1 commonly. |
| PJ3<br>INTLV | 1 | Input/Output<br>Input | Port J3: Programmable as input or output<br>Interleave Mode: The TMP1962F10AXBG enters Interleave mode when this pin is sampled high (DVCC21) at the rising edge of RESET. During a reset sequence, this pin should be pulled up to a logic 1. |
| PJ4<br>ENDIAN | 1 | Input/Output<br>Input | Port J4: Programmable as input or output<br>This pin is used to set the mode. If this pin is sampled high (DVCC21) at the rising edge of RESET, the TMP1962F10AXBG enters Big-endian mode. If this pin is sampled low at the rising edge of RESET, the TMP1962F10AXBG enters Little-endian mode. During a reset sequence, this pin should be pulled up to a logic 1or pulled down to a logic 0 depending the endian to be used. |
| PK0-PK7<br>KEY0-KEY7 | 8 | Input/Output<br>Input | Port K: Programmable as input or output<br>Key on wake up input 0-7 (dynamic pull-up selectable) (with internal pull-up register)<br>Schmitt trigger input |
| PL0<br>TA4IN | 1 | Input/Output<br>Input | Port L0: Programmable as input or output<br>8-bit Timer 4 Input: Input to 8-bit Timer 4 |
| PL1<br>TA6IN | 1 | Input/Output<br>Input | Port L1: Programmable as input or output<br>8-bit Timer 6 Input: Input to 8-bit Timer 6 |
| PL2<br>TA8IN | 1 | Input/Output<br>Input | Port L2: Programmable as input or output<br>8-bit Timer 8 Input: Input to 8-bit Timer 8 |
| PL3<br>TAAIN | 1 | Input/Output<br>Input | Port L3: Programmable as input or output<br>8-bit Timer A Input: Input to 8-bit Timer A |
| PL4<br>TB0IN0 | 1 | Input/Output<br>Input | Port L4: Programmable as input or output<br>16-Bit Timer 0 Input 0: Count/capture trigger input to 16-bit Timer 0 |
| PL5<br>TB0IN1 | 1 | Input/Output<br>Input | Port L5: Programmable as input or output<br>16-Bit Timer 0 Input 1: Capture trigger input to 16-bit Timer 0 |
| PL6<br>TB1IN0 | 1 | Input/Output<br>Input | Port L6: Programmable as input or output<br>16-Bit Timer 1 Input 0: Count/capture trigger input to 16-bit Timer 1 |
| PL7<br>TB1IN1 | 1 | Input/Output<br>Input | Port L7: Programmable as input or output<br>16-Bit Timer 1 Input 1: Capture trigger input to 16-bit Timer 1 |
| PM0-PM7 | 8 | Input/Output | Port M: Individually programmable input or output |
| PN0<br>TXD6 | 1 | Input/Output<br>Output | Port N0: Programmable as input or output<br>Serial Transmit Data 6: Programmable as a push-pull or open-drain output |
| PN1<br>RXD6 | 1 | Input/Output<br>Input | Port N1: Programmable as input or output<br>Serial Receive Data 6 |
| PN2<br>SCLK6<br>$\overline{\text{CTS6}}$ | 1 | Input/Output<br>Input<br>Input | Port N2: Programmable as input or output<br>Serial Clock Input/Output 6<br>Serial Clear-to-Send 6<br>Programmable as a push-pull or open-drain output |
| PN3-PN7 | 5 | Input/Output | Port N3-N7: Individually programmable input or output |
| PO0-PO7 | 8 | Input/Output | Port O: Individually programmable input or output |
| PP0-PP7 | 8 | Input/Output | Port P: Individually programmable input or output |

Table 2.2 Pin Names and Function (6/6)

| Pin Name | # of Pins | Type | Function |
|---|---|---|---|
| $\overline{NMI}$ | 1 | Input | Nonmaskable Interrupt Request: Causes an NMI interrupt on the falling edge |
| $\overline{PLLOFF}$ | 1 | Input | This pin should be tied to High (DVCC21) when the frequency multiplied clock from the PLL is used; Otherwise, it should be tied to Low.<br>Schmitt trigger input |
| RSTPUP | 1 | Input | During a reset sequence, Port 3 and Port 4 are pull-up enabled if this pin is sampled High (DVCC32), and disabled if this pin is sampled Low.<br>Schmitt trigger input |
| $\overline{RESET}$ | 1 | Input | Reset (with internal pull-up register): Initializes the whole TMP1962F10AXBG<br>Schmitt trigger input |
| X1/X2 | 2 | Input/Output | Connection pins for a high-speed crystal |
| XT1/XT2 | 2 | Input/Output | This pin should be left open. |
| $\overline{DRESET}$ | 1 | Input | Debug Reset: Signal for DSU- ICE (Schmitt trigger input, with internal pull-up register) |
| DCLK | 1 | Output | Debug Clock: Signal for DSU-ICE |
| $\overline{DBGE}$ | 1 | Input | Debug Enable: Signal for DSU-ICE (Schmitt trigger input, with internal pull-up register) |
| PCST3-0 | 4 | Output | PC Trace Status: Signal for DSU-ICE |
| SDI/ $\overline{DINT}$ | 1 | Input | Serial Data Input/Debug Interrupt: Signal for DSU-ICE<br>(Schmitt trigger input, with internal pull-up register) |
| SDAO/TPC | 1 | Output | Serial Data and Address Output/Target PC: Signal for DSU-ICE |
| TCK | 1 | Input | Test Clock Input: Signal for JTAG test (Schmitt trigger input, with internal pull-up register) |
| TMS | 1 | Input | Test Mode Select Input: Signal for JTAG test (Schmitt trigger input, with internal pull-up register) |
| TDI | 1 | Input | Test Data Input: Signal for JTAG test (Schmitt trigger input, with internal pull-up register) |
| TDO | 1 | Output | Test Data Output: Signal for JTAG test |
| $\overline{TRST}$ | 1 | Input | Test Reset Input: Signal for JTAG test (Schmitt trigger input, with internal pull-up register) |
| BW0-1 | 2 | Input | Both BW0 and BW1 should be tied to High (DVCC21). (Schmitt trigger input) |
| VREFH | 1 | Input | Input pin for high reference voltage for the A/D Converter<br>This pin should be connected to the AVCC pin when the A/D Converter is not used. |
| VREFL | 1 | Input | Input pin for low reference voltage for the A/D Converter<br>This pin should be connected to the AVCC pin when the A/D Converter is not used. |
| AVCC31-32 | 2 | — | Power supply pins for the A/D Converter. These pins should always be connected to power supply even when the A/D Converter is not used. |
| AVSS | 3 | — | Ground pins for the A/D Converter. These pins should always be connected to ground even when the A/D Converter is not used. |
| TEST0 | 1 | — | Test pin: This pin should be left open or tied to ground. |
| TEST1 | 1 | Input | Test pin: This pin should be tied to ground. |
| TEST2 | 1 | — | Test pin: This pin should be left open or tied to ground. |
| TEST3 | 1 | — | Test pin: This pin should be left open or tied to ground. |
| TEST4 | 1 | — | Test pin: This pin should be left open or tied to ground. |
| TEST5 | 1 | Input | Test pin: This pin should be tied to ground. |
| SYSRDY | 1 | Output | Signal to grant access to a flash memory |
| CVCC2 | 1 | — | Power supply pins for an oscillator: 2.5 V |
| CVSS | 1 | — | Ground pin for an oscillator (0 V) |
| CVCCH | 1 | — | This pin should be left open. |
| CAP1 | 1 | — | This pin should be left open. |
| CAP2 | 1 | — | This pin should be left open. |
| FVCC2 | 3 | — | Power supply pins for a Flash memory: 2.5 V |
| FVCC3 | 1 | — | Power supply pin for a flash memory: 3 V |
| FVSS | 4 | — | Ground pins for a flash memory (0 V) |
| DVCC21-22 | 5 | — | Power supply pins: 2.5 V |
| DVCC31-34 | 9 | — | Power supply pins: 3 V |
| DVSS | 9 | — | Ground pins (0V) |

**Note 1:** **PJ1, PJ2, PJ3 and PJ4 should be held at the prescribed logic states for one system clock cycle before and after the rising edge of RESET, with the RESET signal being stable in either logic state.**

**Note 2:** **Debugging with a DSU-probe is enabled on the TMP1962F10AXBG. Connection to the DSU-probe is available on the TMP1962C10BXBG, but a DSU-probe can not read the contents of on-chip ROM or write to registers other than the processor core, on-chip memory and external device.**

Table 2.3 shows correspondence between pins and power supply pins.

Table 2.3 Pins and Corresponding Power Supply Pins

| Pin | Power Supply | | Pin | Power Supply | |
|---|---|---|---|---|---|
| | Mask Type | Flash Type | | Mask Type | Flash Type |
| P0 | DVCC33 | DVCC33 | PO | DVCC31 | DVCC31 |
| P1 | DVCC33 | DVCC33 | PP | DVCC31 | DVCC31 |
| P2 | DVCC33 | DVCC33 | X1 | CVCC15 | CVCC2 |
| P3 | DVCC33 | DVCC33 | X2 | CVCC15 | CVCC2 |
| P4 | DVCC33 | DVCC33 | $\overline{\text{RESET}}$ | DVCC2 | DVCC21 |
| P5 | DVCC33 | DVCC33 | $\overline{\text{NMI}}$ | DVCC2 | DVCC21 |
| P6 | DVCC33 | DVCC33 | $\overline{\text{PLLOFF}}$ | DVCC2 | DVCC21 |
| P7 | AVCC32 | AVCC32 | $\overline{\text{DRESET}}$ | DVCC2 | DVCC21 |
| P8 | AVCC32 | AVCC32 | DCLK | DVCC2 | DVCC21 |
| P9 | AVCC31 | AVCC31 | $\overline{\text{DBGE}}$ | DVCC2 | DVCC21 |
| PA | DVCC32 | DVCC32 | PCST3-0 | DVCC2 | DVCC21 |
| PB | DVCC32 | DVCC32 | SDI/ $\overline{\text{DINT}}$ | DVCC2 | DVCC21 |
| PC | DVCC32 | DVCC32 | SDAO/TPC | DVCC2 | DVCC21 |
| PD | DVCC32 | DVCC32 | TCK | DVCC34 | DVCC34 |
| PE | DVCC32 | DVCC32 | TMS | DVCC34 | DVCC34 |
| PF | DVCC32 | DVCC32 | TDI | DVCC34 | DVCC34 |
| PG | DVCC32 | DVCC32 | TDO | DVCC34 | DVCC34 |
| PH | DVCC32 | DVCC32 | $\overline{\text{TRST}}$ | DVCC34 | DVCC34 |
| PI | DVCC34 | DVCC34 | BW1-0 | DVCC2 | DVCC21 |
| PJ | DVCC2 | DVCC21 | RSTPUP | DVCC32 | DVCC32 |
| PK | DVCC34 | DVCC34 | | | |
| PL | DVCC34 | DVCC34 | | | |
| PM | DVCC34 | DVCC34 | | | |
| PN | DVCC31 | DVCC31 | | | |

Table 2.4 shows the supply voltage for power supply pins.

Table 2.4 Supply Voltage for Power Supply Pins

| Power Supply Pin | Supply Voltage | Applied for |
|---|---|---|
| DVCC15 | 1.35 V - 1.65 V | Mask Type |
| CVCC15 | 1.35 V - 1.65 V | |
| DVCC2 | 2.3 V - 3.3 V | |
| DVCC21 | 2.2 V - 2.7 V | Flash Type |
| DVCC22 | 2.2 V - 2.7 V | |
| CVCC2 | 2.2 V - 2.7 V | |
| FVCC2 | 2.2 V - 2.7 V | |
| FVCC3 | 2.9 V - 3.6 V | |
| DVCC31 - 34 | 1.65 V - 3.3 V | Mask/Flash Type |
| AVCC31 - 32 | 2.7 V - 3.3 V | |

**Note 1: AVCC32 ≤ AVCC31**

- **When P7 to P9 are used as A/D converter inputs: 2.7 ≤ AVCC3***
- **When P9 (powered by AVCC31) is used as an A/D converter input while P7 and P8 (powered by AVCC32) are used as ports:**

  **2.7 V ≤ AVCC31 ≤ 3.3 V**

  **1.65 V ≤ AVCC32 ≤ AVCC31**
- **When P7 (powered by AVCC32) is used as an A/D converter input which P8 (powered by AVCC32) and P9 (powered by AVCC31) are used as ports:**

  **2.7 V ≤ AVCC32 ≤ AVCC31 ≤ 3.3 V**

**Note2: With power supplies for CPU and internal logic (mask type: DVCC15/DVCC2/DVCC15, and flash type: DVCC21/DVCC22/CVCC2/FVCC2/FVCC3) being applied, power supplies for other I/O ports can be interrupted on the TMP1962. However, when AVCC31 for analog power supply is interrupted, overlap current is generated on the TMP1962F10A with on-chip flash memory during the transition to be stable in 0 V. Overlap current can be suppressed by AD conversion of the conversion result 0 V before interrupting AVCC31 power supply, but suppress it on devices.**

# 3. Flash Memory

This chapter describes the flash memory of the TMP1962F10AXBG, a flash version of the TMP1962C10BXBG. The TMP1962F10AXBG contains a 1-Mbyte flash EEPROM and 40-kbyte RAM whereas the TMP1962C10BXBG contains a 1-Mbyte ROM and a 40-kbyte RAM. In other respects, the hardware configuration and the functionality of the TMP1962F10AXBG are identical to those of the TMP1962C10BXBG. For descriptions of the on-chip I/O peripherals, refer to the TMP1962C10BXBG datasheet.

## 3.1 Features

(1) Organization

The TMP1962F10AXBG contains 8 Mbits (1024 kbytes) of flash memory, which is divided into a total of 8 blocks (128-kbyte x 8) to allow for independent protection from program and erase for each block. While the CPU can access information in the flash through a full 32-bit data bus, an external flash programmer can only perform 16-bit data bus writes to the flash.

(2) Access Types

The flash memory of the TMP1962F10AXBG provides the interleaved access type.

(3) Program/Erase Time

- Chip programming time: 15 seconds (typ.) including verify operations
- Chip erase time: 40 seconds (typ.), including verify operations

**Note: These program and erase times are typical values and do not include data transfer overhead. The actual chip program and erase times depend on the programming method used.**

(4) Programming Modes

Several options exist to program the TMP1962F10AXBG flash memory. On-Board Programming modes allow for re-programming of the flash memory while the chip is soldered on a printed circuit board. Programmer mode utilizes an EPROM programmer to perform code updates.

- On-Board Programming modes
  1) User Boot mode

     Supports use of a user-written programming algorithm.
  2) Single Boot mode

     Downloads the new program code using a Toshiba-defined serial interface protocol.
- Programmer Mode

  Supports use of a general-purpose EPROM programmer.

(5) Re-programming

The TMP1962F10AXBG flash memory is compatible with the JEDEC standards, except a few unique functions. Thus, it is easy to migrate from a discrete flash memory device to the on-chip flash memory of the TMP1962F10AXBG. The TMP1962F10AXBG contains hardware to perform programming and erase operations automatically. This eliminates the need for the user to code complex program and erase sequences.

The security feature of the TMP1962F10AXBG flash memory prevents the stored data from being read while it is being re-programmed with programming equipment. The TMP1962F10AXBG also allows the user to protect individual blocks of the flash memory against program or erase through software commands; however, 12-V VPP programming does not support data protection on a block-by-block basis. The flash memory is secured automatically by all of 8 blocks being protected. Unsecuring the flash memory automatically erases the stored data prior to unprotecting the blocks.

| JEDEC Standard | Changes and Enhancements |
|---|---|
| Auto Program | Added feature: Security Auto Program |
| Auto Chip Erase | Changed feature: Block protection is available only under software control. |
| Auto Block Erase | Removed feature: Erase Resume/Suspend mode |
| Auto Multi-Block Erase | |
| Data Polling / Toggle Bit | |

## 3.2 Block Diagram

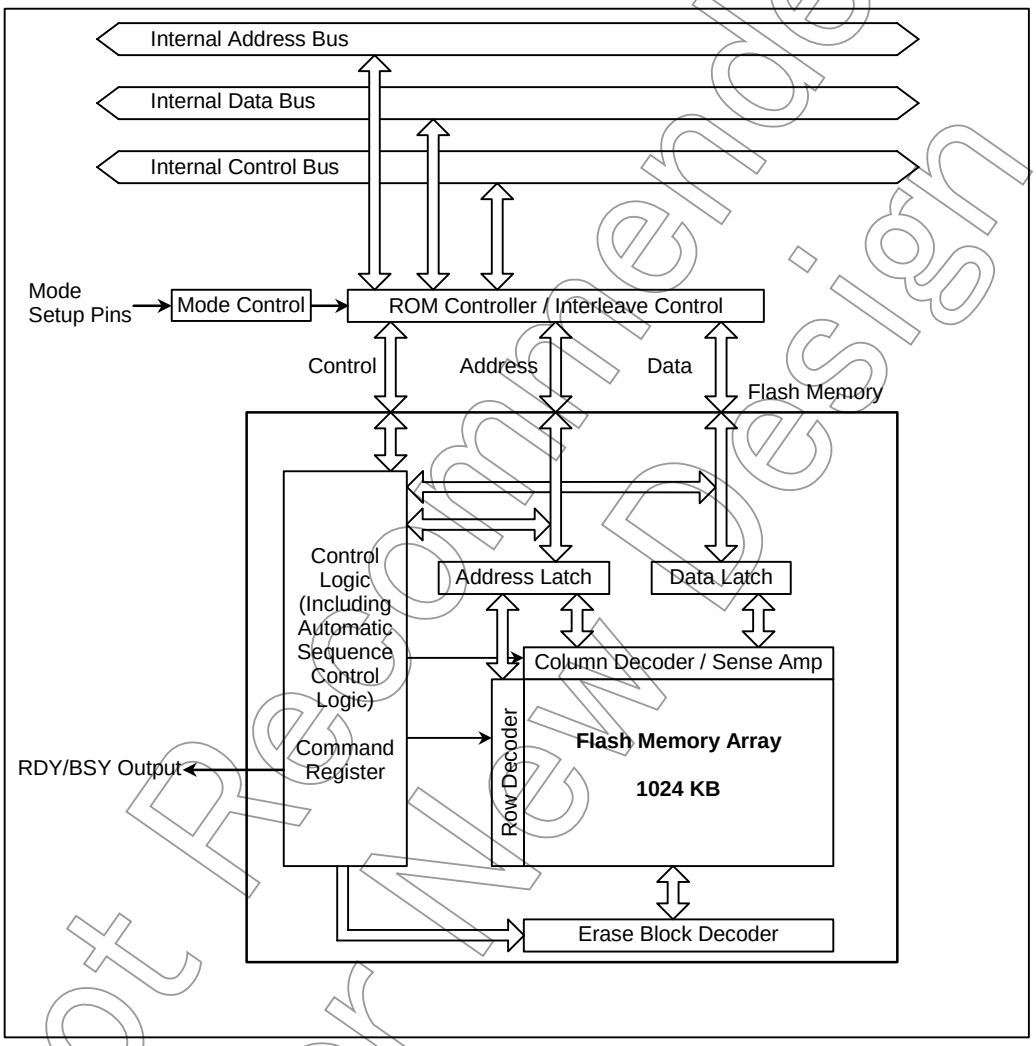



Figure 3.1 Flash Memory Block Diagram

## 3.3 Operating Modes

### 3.3.1 Overview

The TMP1962F10AXBG offers a total of five operating modes, including the one in which the flash memory is unused.

Table 3.1 Operating Modes

| Operating Mode | | Description |
|---|---|---|
| Single-Chip Mode | | After a reset, the TX19 core processor executes out of the on-chip flash memory. Set the INTLV pin to High level when $\overline{\text{RESET}}$ is released. |
| | Normal Mode | Single-Chip mode is further divided into Normal mode in which the user application executes and User Boot mode which allows for re-programming of the flash memory while the TMP1962F10AXBG is installed on a printed circuit board.<br><br>The user can freely define how to switch between Normal mode and User Boot mode. For example, the logic state on, say, Port 00, can be used to determine whether to put the flash memory in Normal mode or User Boot mode. The user must include a routine in the application program to test the state of that port. |
| | User Boot Mode | |
| Single Boot Mode | | After a reset, the TX19 core processor executes out of the on-chip boot ROM (which is a mask ROM). The boot ROM contains a routine to aid users in performing on-board programming of the flash memory via a serial port of the TMP1962F10AXBG. The serial port is connected to an external host which transfers new data according to a prescribed protocol. |
| Programmer Mode | | This mode allows re-programming of the flash memory with a general-purpose EPROM programmer. Use the programmer and programming adaptor recommended by Toshiba. |

The on-chip flash memory can be re-programmed in one of the following three modes: User Boot mode, Single Boot mode and Programmer mode. Of these modes, User Boot mode and Single Boot mode are collectively referred to as on-board programming modes.

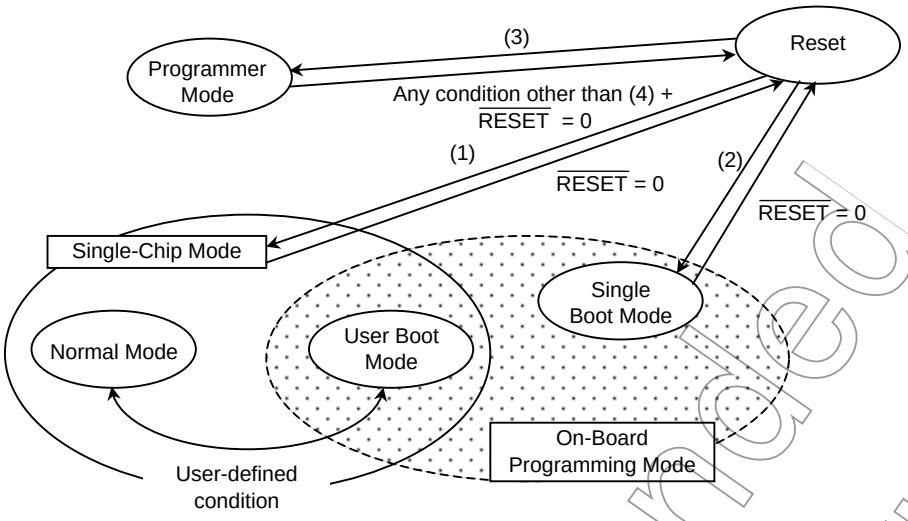
The logic states on the BW0, BW1, $\overline{\text{BOOT}}$ and INTLV pins during a reset sequence determine the mode of operation for the flash memory, as shown in Table 3.2. After $\overline{\text{RESET}}$ is released, PJ2 ($\overline{\text{BOOT}}$) and PA2 (INTLV) can be configured as general-purpose I/O pins.

After a reset, the CPU operates in compliance with the selected mode, except for Programmer mode. When Programmer mode is selected, $\overline{\text{RESET}}$ must be held at logic 0. The input pins listed in Table 3.2 must remain stable once the flash memory is put in a given mode of operation.

Table 3.2 Modes of Operation

| # | Operating Mode | Input Pins | | | | |
|---|---|---|---|---|---|---|
| | | $\overline{\text{RESET}}$ | BW0 | BW1 | $\overline{\text{RESET}}$ | INTLV |
| (1) | Single-Chip Mode | 0 → 1 | 1 | 1 | 1 | 1 |
| (2) | Single Boot Mode | 0 → 1 | 1 | 1 | 0 | Note 1 |
| (3) | Programmer Mode | 0 | 0 | 1 | Note 1 | Note 1 |

**Note 1: Don't care. The pins must be held at 1 or 0, however.**




Parenthesized numbers indicate that the relevant pins are at the logic states shown in Table 3.2.

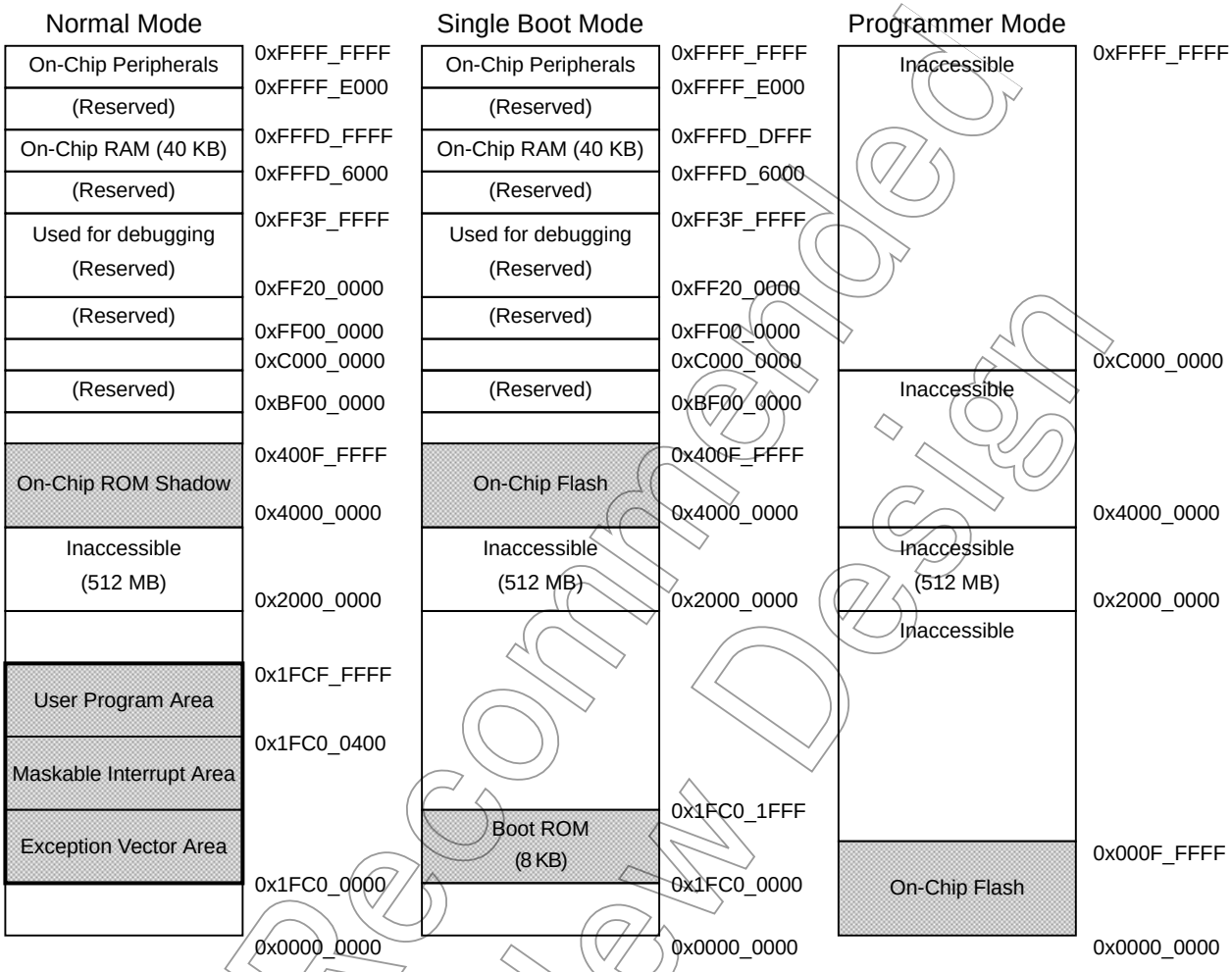
Figure 3.2 Mode Transitions

### 3.3.2 Reset Operation

To reset the TMP1962F10AXBG, $\overline{\text{RESET}}$ must be asserted for at least 12 system clock periods after the power supply voltage and the internal high-frequency oscillator have stabilized. This time is typically 2.37 μs at 40.5 MHz when the on-chip PLL is utilized.

### 3.3.3 Memory Maps

The memory map for the TMP1962F10AXBG varies according to the operation mode selected for the on-chip flash memory. Following are the memory maps in each operation mode.




Note: The addresses shown above are physical addresses.

Figure 3.3 TMP1962F10AXBG Memory Maps

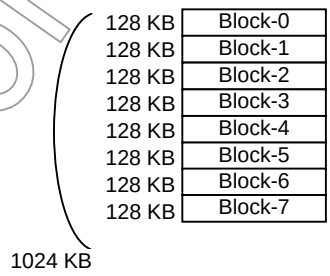



Figure 3.4 Flash Memory Block Architecture

Table 3.3 Block Addresses

| | User Boot Mode | Boot Mode | Programmer Mode |
|---|---|---|---|
| Block-0 | 0x1FC0_0000 - 0x1FC1_FFFF<br>(or 0x4000_0000 - 0x4001_FFFF) | 0x1FC0_0000 - 0x1FC1_FFFF | 0x0000_0000 - 0x0001_FFFF |
| Block-1 | 0x1FC2_0000 - 0x1FC3_FFFF<br>(or 0x4002_0000 - 0x4003_FFFF) | 0x1FC2_0000 - 0x1FC3_FFFF | 0x0000_8000 - 0x0003_FFFF |
| Block-2 | 0x1FC4_0000 - 0x1FC5_FFFF<br>(or 0x40040000 - 0x4005_FFFF) | 0x1FC4_0000 - 0x1FC5_FFFF | 0x0001_0000 - 0x0005_FFFF |
| Block-3 | 0x1FC6_0000 - 0x1FC7_FFFF<br>(or 0x4006_0000 - 0x4007_FFFF) | 0x1FC6_0000 - 0x1FC7_FFFF | 0x0001_8000 - 0x0007_FFFF |
| Block-4 | 0x1FC8_0000 - 0x1FC9_FFFF<br>(or 0x4008_0000 - 0x4009_FFFF) | 0x1FC8_0000 - 0x1FC9_FFFF | 0x0002_0000 - 0x0009_FFFF |
| Block-5 | 0x1FCA_0000 - 0x1FCB_FFFF<br>(or 0x400A_0000 - 0x400B_FFFF) | 0x1FCA_0000 - 0x1FCB_FFFF | 0x0002_8000 - 0x000A_FFFF |
| Block-6 | 0x1FCC_0000 - 0x1FCD_FFFF<br>(or 0x400C_0000 - 0x400D_FFFF) | 0x1FCC_0000 - 0x1FCD_FFFF | 0x0003_0000 - 0x000B_FFFF |
| Block-7 | 0x1FCE_0000 - 0x1FCF_FFFF<br>(or 0x400E_0000 - 0x400F_FFFF) | 0x1FCE_0000 - 0x1FCF_FFFF | 0x0003_8000 - 0x000C_FFFF |

### 3.3.4 Interleave Mode

If J3 (PJ3) is sampled high at the rising edge of $\overline{\text{RESET}}$ , the flash memory enters Interleave mode. The flash memory must be configured into Interleave mode.

### 3.3.5 Block Protection

The TMP1962F10AXBG flash memory is organized into a total of 8 blocks (128 kbyte× 8). To protect stored data from any program and erase operations, each block has a protect bit, which can be set by executing the Block Protect command sequence. Blocks in protection mode are protected from even the Chip Erase and Multi-Block Erase commands; these commands erase only unprotected blocks. Since protection status is stored in flash memory cells, it is retained if the chip is powered off. When all blocks are protected, the data stored in these blocks are protected from being read in Programmer mode, which provides a security feature.

### 3.3.6 DSU-probe Interface

The DSU-probe interface is used for software debugging using an external DSU-probe unit. This serves as an interface to the DSU-probe, and can not be used as general-purpose port. Consult the DSU-probe operation manual for a description of debugging using theDSU-probe. When the TMP1962F10AXBG is in DSU mode, the on-chip flash memory provides a security feature.

(1) Flash security feature

The TMP1962F10AXBG supports on-board debugging while it is installed on a printed circuit board. The TMP1962F10AXBG provides a security feature to prevent intrusive access to the flash memory. When the flash memory is in the secure state, a DSU-probe is denied access to the entirety of the flash memory.

(2) Securing the flash (Disabling debugging with a DSU-probe)

Once program debug is completed, write the Protect command to all of 8 blocks. This turns on the flash security feature. While the flash memory is in the secure state, a DSU-probe can not read its contents. When the chip is powered off and powered on again, the flash memory is secured, which disables debugging using a DSU-probe until the flash memory is unsecured.

(3) Unsecuring the flash (Enabling debugging with a DSU-probe)

The flash memory may only be unsecured by clearing the SEQON bit in the SEQMOD register and then writing a special code (0x0000_00C5) to the Security Control (SEQCNT) register. This prevents runaway software from inadvertently turning off the security feature. Unsecuring the flash memory enables the DSU interface. The flash memory can be secured again by setting the SEQON bit in the SEQMOD and writing 0x0000_00C5 to the SEQCNT while the chip is powered.

SEQMOD (0xFFFF_E510)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | — | — | — | — | — | — | — | SEQON |
| Read/Write | — | — | — | — | — | — | — | R/W |
| Reset Value | — | — | — | — | — | — | — | 1 |
| Function | | | | | | | | 1: Security on<br>0: Security off |

**Note: This register must be read as a 32-bit quantity. Bits 1 to 31 are read as 0s.**

SEQCNT (0xFFFF_E514)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | | | | | |
| Read/Write | W | | | | | | | |
| Reset Value | | | | | | | | |
| Function | Must be written as 0x0000_00C5. | | | | | | | |
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Name | | | | | | | | |
| Read/Write | W | | | | | | | |
| Reset Value | | | | | | | | |
| Function | Must be written as 0x0000_00C5. | | | | | | | |
| | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Name | | | | | | | | |
| Read/Write | W | | | | | | | |
| Reset Value | | | | | | | | |
| Function | Must be written as 0x0000_00C5. | | | | | | | |
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Name | | | | | | | | |
| Read/Write | W | | | | | | | |
| Reset Value | | | | | | | | |
| Function | Must be written as 0x0000_00C5. | | | | | | | |

**Note 1: This register is read as a 32-bit quantity.**
**Note 2: The security feature of the TMP1962F10AXBG flash memory is not intended to guarantee rigid security protection. In cases where security protection is of utmost importance, use the TMP1962C10BXBG that contains mask ROM.**

(4) Application example

The following flowchart exemplifies how to use the security feature with a DSU-ICE.

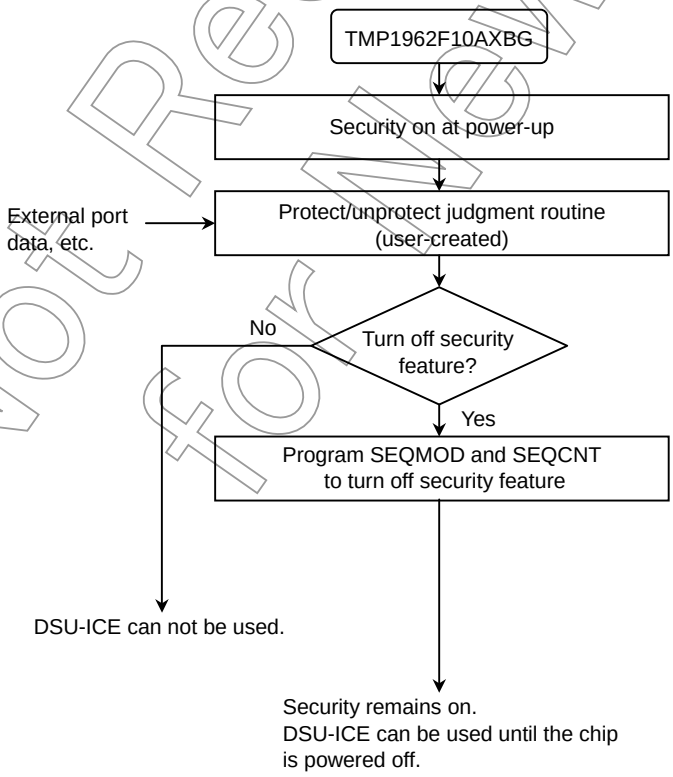



Figure 3.5 Using the Security Feature

## 3.4 On-Board Programming Mode

On-board programming modes allo

w for re-programming of the flash memory while the TMP1962F10AXBG is soldered on a printed circuit board. In Single Boot mode, new data comes from a serial port under control of a Toshiba-provided routine in the boot ROM. User Boot mode allows you to create an algorithm of your own for flash memory erase and program operations.

The TMP1962F10AXBG flash memory provides a security feature to prevent intrusive access to the flash memory while in Programmer mode. This security feature can be enabled upon completion of on-board programming to reduce the potential risk of software leaks to third parties.

### 3.4.1 User Boot Mode (Single-Chip Mode)

User Boot mode allows you to create a programming algorithm of your own. This mode supports situations where the flash memory is to be re-programmed via a bus other than serial I/O. User Boot mode is one of the two submodes in Single-Chip mode; the other submode is Normal mode in which the CPU executes the user application. To re-program the flash memory, the mode of operation must be switched from Normal mode to User Boot mode. The user application code must include a mode judgment routine as part of the reset procedure.

The user must define the conditions for mode switching, based on the logic states on I/O ports of the TMP1962F10AXBG. Additionally, the user must incorporate a programming algorithm into the user application code that is to be executed after User Boot mode is entered.

It is not possible to read from the flash memory while it is being erased or programmed; therefore, the programming algorithm must be placed and executed outside of the flash memory.

Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption.

All interrupts including the nonmaskable (NMI) interrupt must be globally disabled while the flash memory is being erased or programmed.

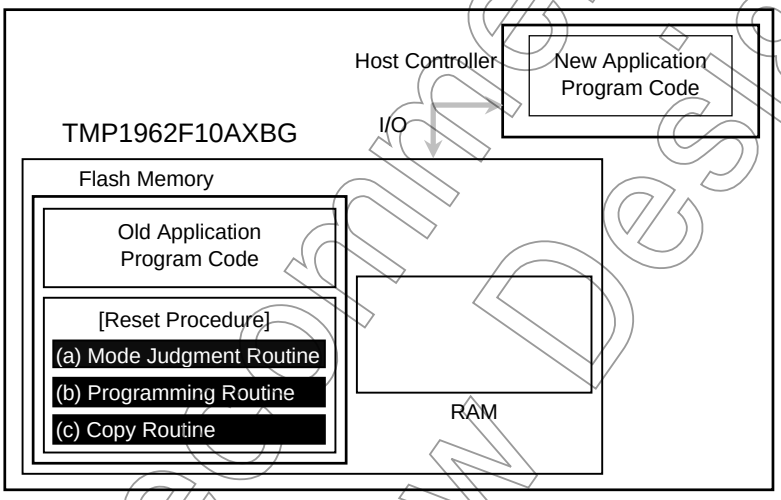
The pages that follow describe the general procedures for two cases where the programming routine is: a) stored within the TMP1962F10AXBG flash memory, and b) loaded from an external controller. For a detailed description of the erase and program sequence, refer to Section 3.6, *On-Board Programming and Erasure.*

User Boot Mode

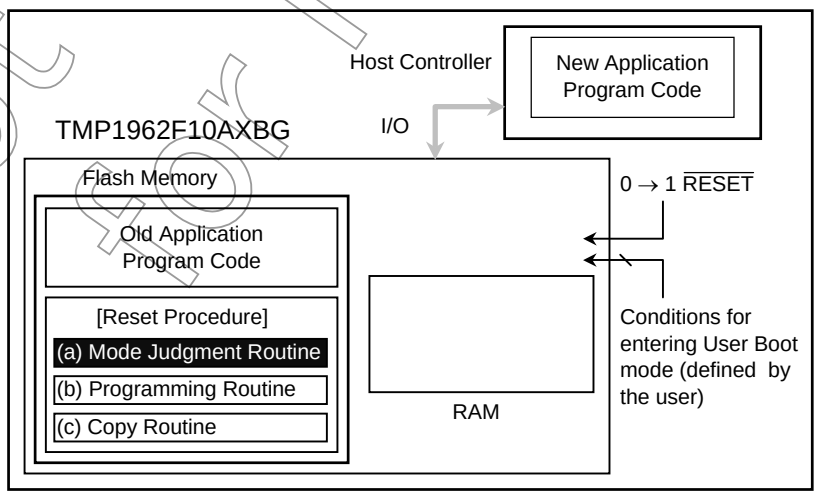
(1-A) Method 1: Storing a Programming Routine in the Flash Memory

(1) Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMP1962F10AXBG on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

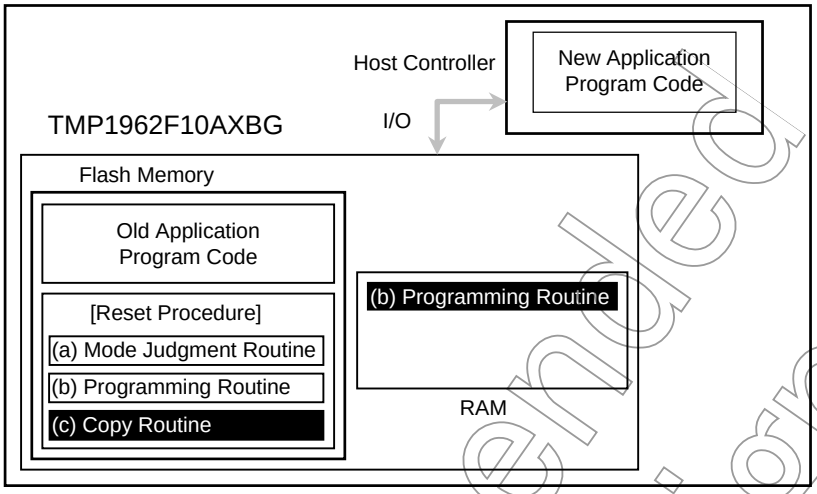
- Mode judgment routine: Code to determine whether or not to switch to User Boot mode
- Programming routine: Code to download new program code from a host controller and re-program the flash memory
- Copy routine: Code to copy the flash programming routine from the TMP1962F10AXBG flash memory to either the TMP1962F10AXBG on-chip RAM or external memory device.



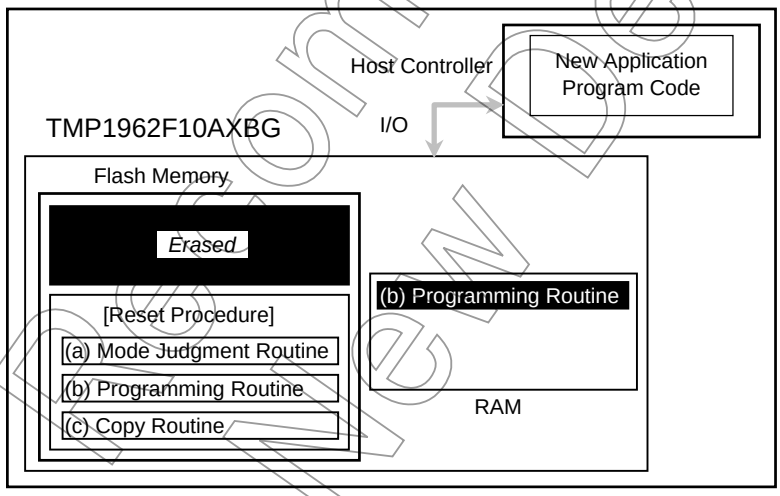

(2) After $\overline{\text{RESET}}$ is released, the reset procedure determines whether to put the TMP1962F10AXBG flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be globally disabled while in User Boot mode.)



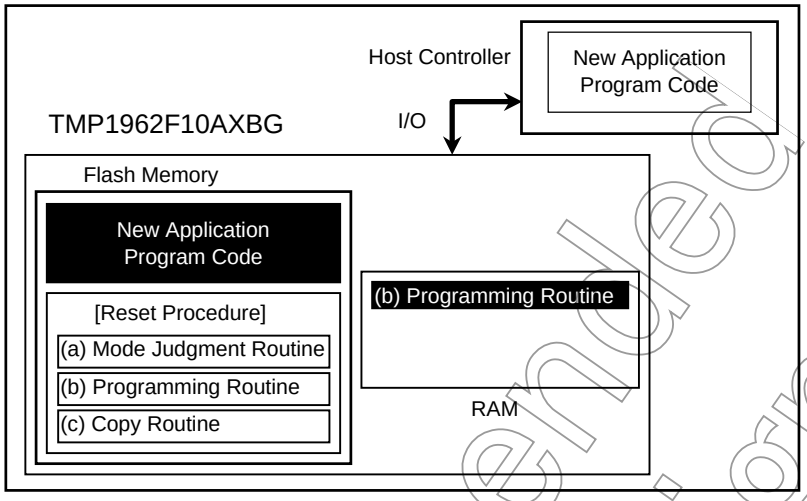
(3) Once User Boot mode is entered, execute the copy routine to copy the flash programming routine to either the TMP1962F10AXBG on-chip RAM or an external memory device. (In the following figure, the on-chip RAM is used.)



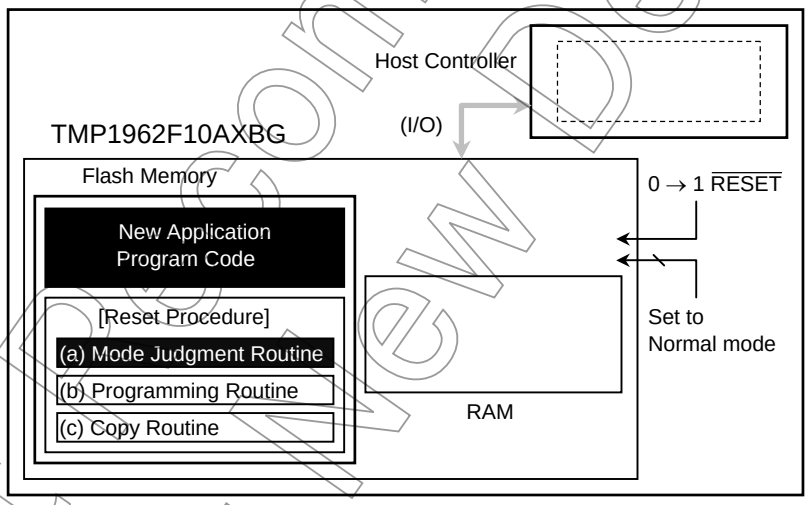

(4) Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code.



(5) Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. Once programming is complete, turn on the protection of that flash block.




(6) Drive $\overline{\text{RESET}}$ low to reset the TMP1962F10AXBG. Upon reset, the on-chip flash memory is put in Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.



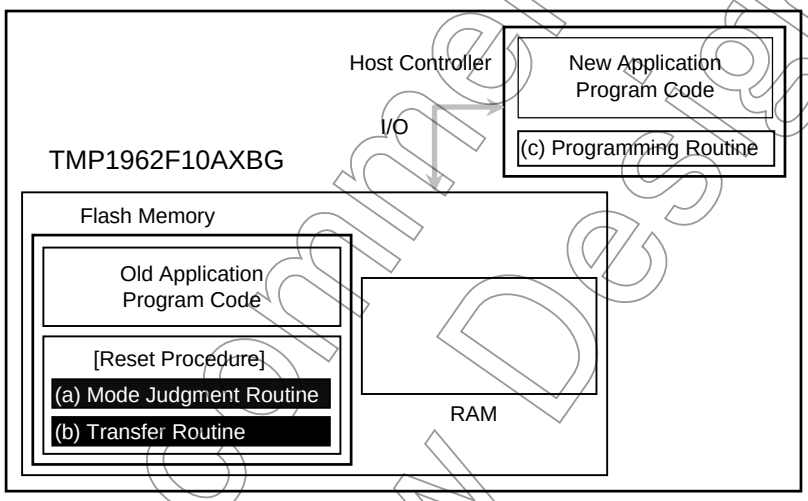
(1-B) Method 2: Transferring a Programming Routine from an External Host

(1) Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMP1962F10AXBG on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

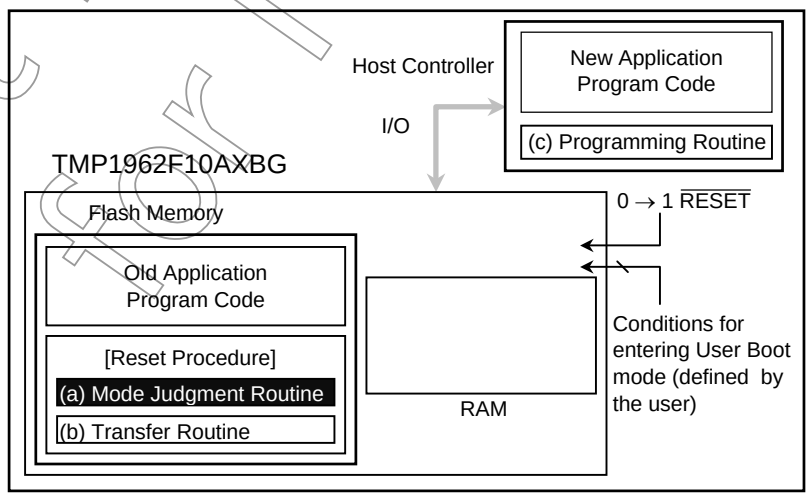
- Mode judgment routine: Code to determine whether or not to switch to User Boot mode
- Transfer routine: Code to download new program code from a host controller

Also, prepare a programming routine on the host controller:

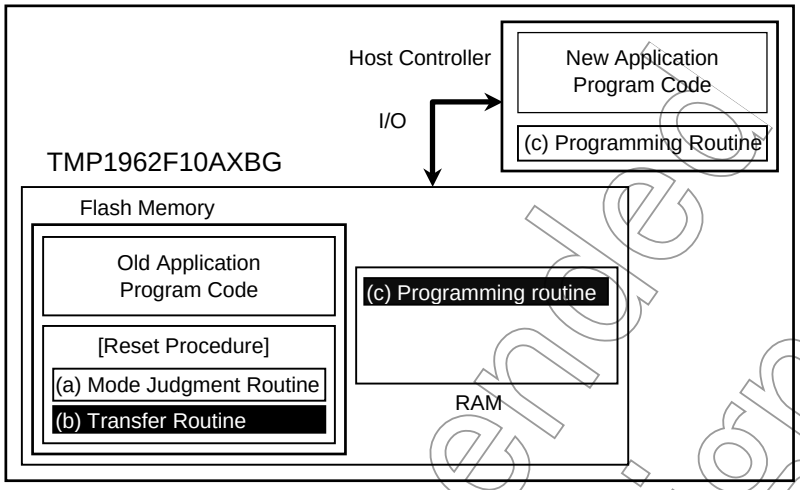
- Programming routine: Code to download new program code from an external host controller and re-program the flash memory



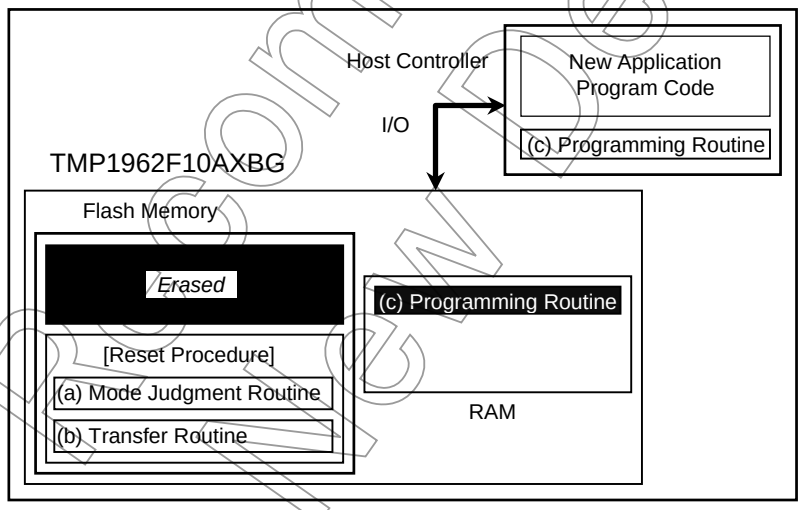

(2) After $\overline{\text{RESET}}$ is released, the reset procedure determines whether to put the TMP1962F10AXBG flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be globally disabled while in User Boot mode.)



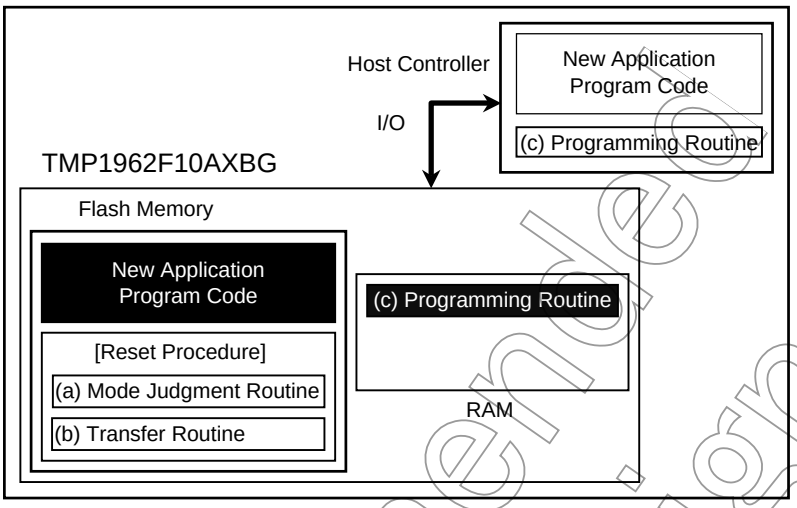
(3) Once User Boot mode is entered, execute the transfer routine to download the flash programming routine from the host controller to either the TMP1962F10AXBG on-chip RAM or an external memory device. (In the following figure, the on-chip RAM is used.)



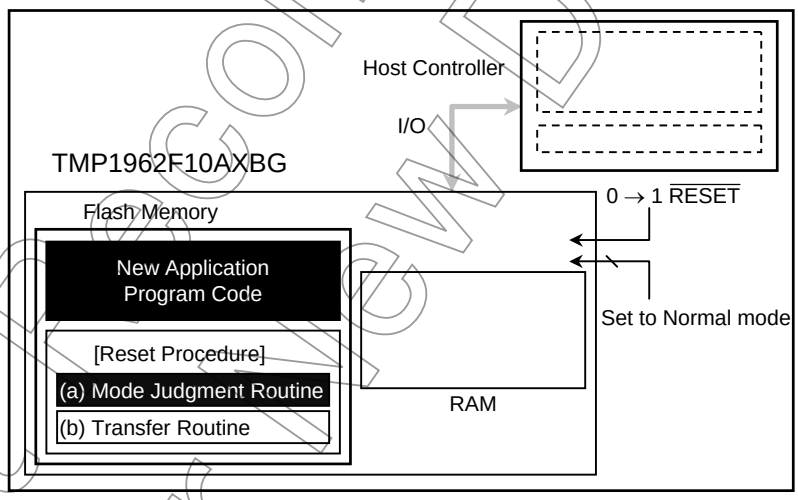

(4) Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code.



(5) Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. Once programming is complete, turn on the protection of that flash block.




(6) Drive $\overline{\text{RESET}}$ low to reset the TMP1962F10AXBG. Upon reset, the on-chip flash memory is put in Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.



## 3.5 Single Boot Mode

In Single Boot mode, the flash memory can be re-programmed by using a program contained in the TMP1962F10AXBG on-chip boot ROM. This boot ROM is a masked ROM. When Single Boot mode is selected upon reset, the boot ROM is mapped to the address region including the interrupt vector table while the flash memory is mapped to an address region different from it (see Figure 3.3 on page 19).

Single Boot mode allows for serial programming of the flash memory. Channel 0 of the SIO (SIO0) of the TMP1962F10AXBG is connected to an external host controller. Via this serial link, a programming routine is downloaded from the host controller to the TMP1962F10AXBG on-chip RAM. Then, the flash memory is re-programmed by executing the programming routine. The host sends out both commands and programming data to re-program the flash memory.

Communications between the SIO0 and the host must follow the protocol described later. To secure the contents of the flash memory, the validity of the application's password is checked before a programming routine is downloaded into the on-chip RAM. If password matching fails, the transfer of a programming routine itself is aborted.

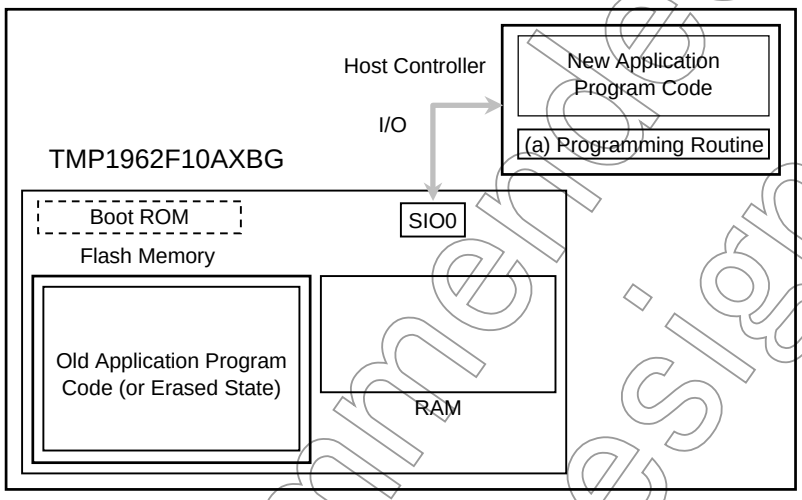
As in the case of User Boot mode, all interrupts including the nonmaskable (NMI) interrupt must be globally disabled in Single Boot mode while the flash memory is being erased or programmed. In Single Boot mode, the boot-ROM programs are executed in Normal mode.

Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations. For a detailed description of the erase and program sequence, refer to 3.4 On-Board Programming and Erasure.

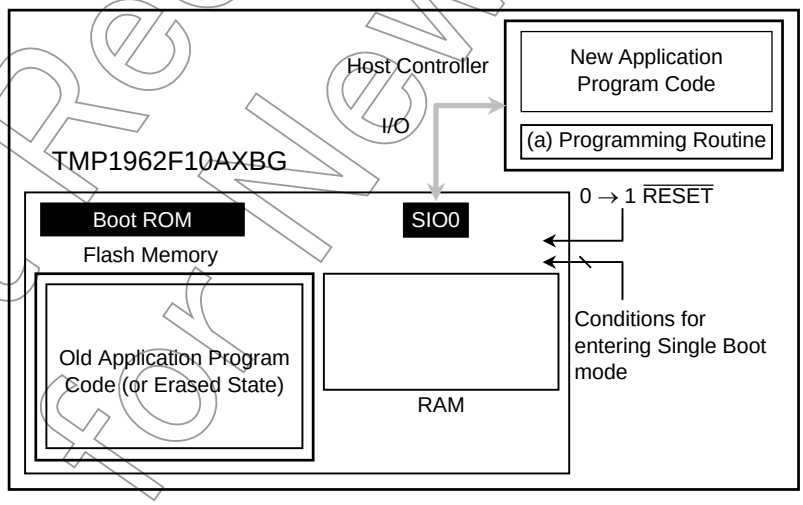
Boot Mode

(2-A) General Procedure: Using the Program in the On-Chip Boot ROM

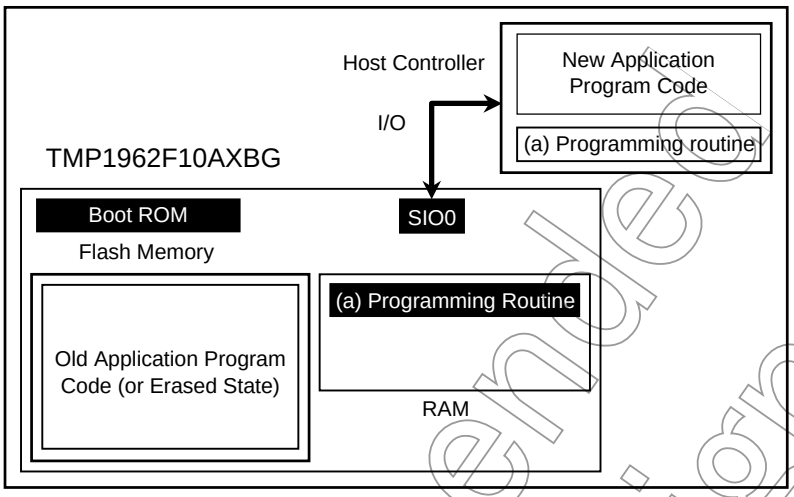
(1) The flash block containing the older version of the program code need not be erased before executing the programming routine. Since a programming routine and programming data are transferred via the SIO0, the SIO0 must be connected to a host controller. Prepare a programming routine on the host controller.



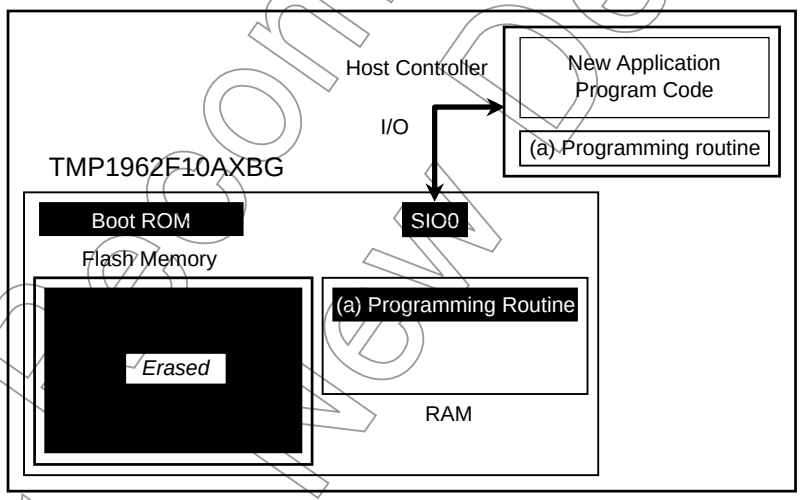

(2) Reset the TMP1962F10AXBG with the mode setting pins held at appropriate logic values, so that the CPU re-boots from the on-chip boot ROM. The 12-byte password transferred from the host controller is first compared to the contents of special flash memory locations. (If the flash block has already been erased, the password is 0xFFFF.)



(3) If the password was correct, the boot program downloads, via the SIO0, the programming routine from the host controller into the on-chip RAM of the TMP1962F10AXBG. The programming routine must be stored in the address range 0xFFFD_6000 – 0xFFFD_EFFF.



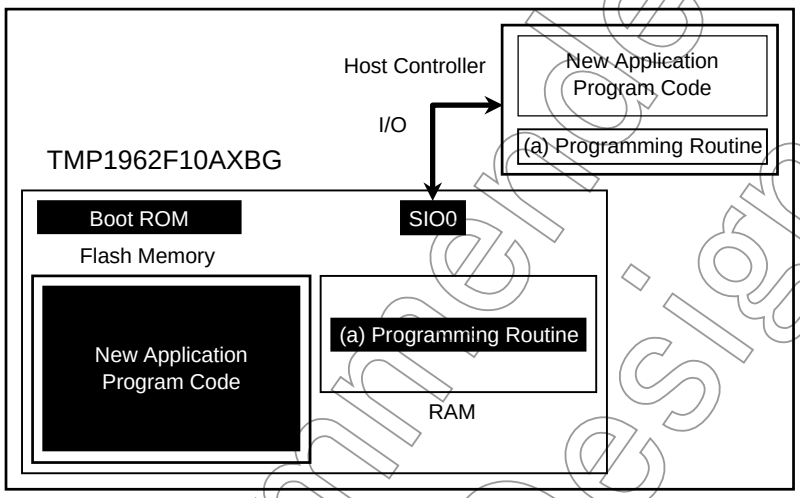

(4) The CPU jumps to the programming routine in the on-chip RAM to erase the flash block containing the old application program code. The Block Erase or Chip Erase command may be used.



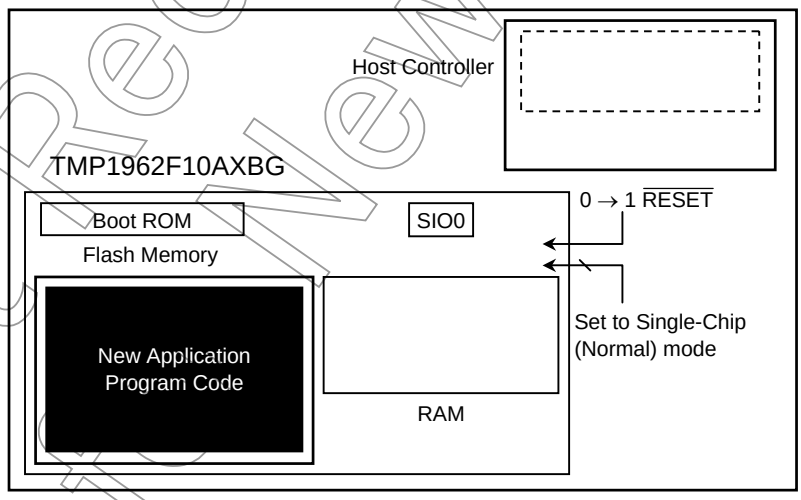
(5) Next, the programming routine downloads new application program code from the host controller and programs it into the erased flash block. Once programming is complete, protection of that flash block is turned on.

It is not allowed to move program control from the programming routine back to the boot ROM.

In the example below, new program code comes from the same host controller via the same SIO channel as for the programming routine. However, once the programming routine has begun to execute, it is free to change the transfer path and the source of the transfer. Create board hardware and a programming routine to suit your particular needs.




(6) When programming of the flash memory is complete, power off the board and disconnect the cable leading from the host to the target board. Turn on the power again so that the TMP1962F10AXBG re-boots in Single-Chip (Normal) mode to execute the new program.



### 3.5.1 Host-to-Target Connection Examples

In Single Boot mode, serial transfer is used to re-program the flash memory while the TMP1962F10AXBG is installed on the board. In this mode, channel 0 of the SIO (SIO0) of the TMP1962F10AXBG is connected to a host controller, which is to issue commands to the target board. Figure 3.6 and Figure 3.7 show examples of host-to-target connections.

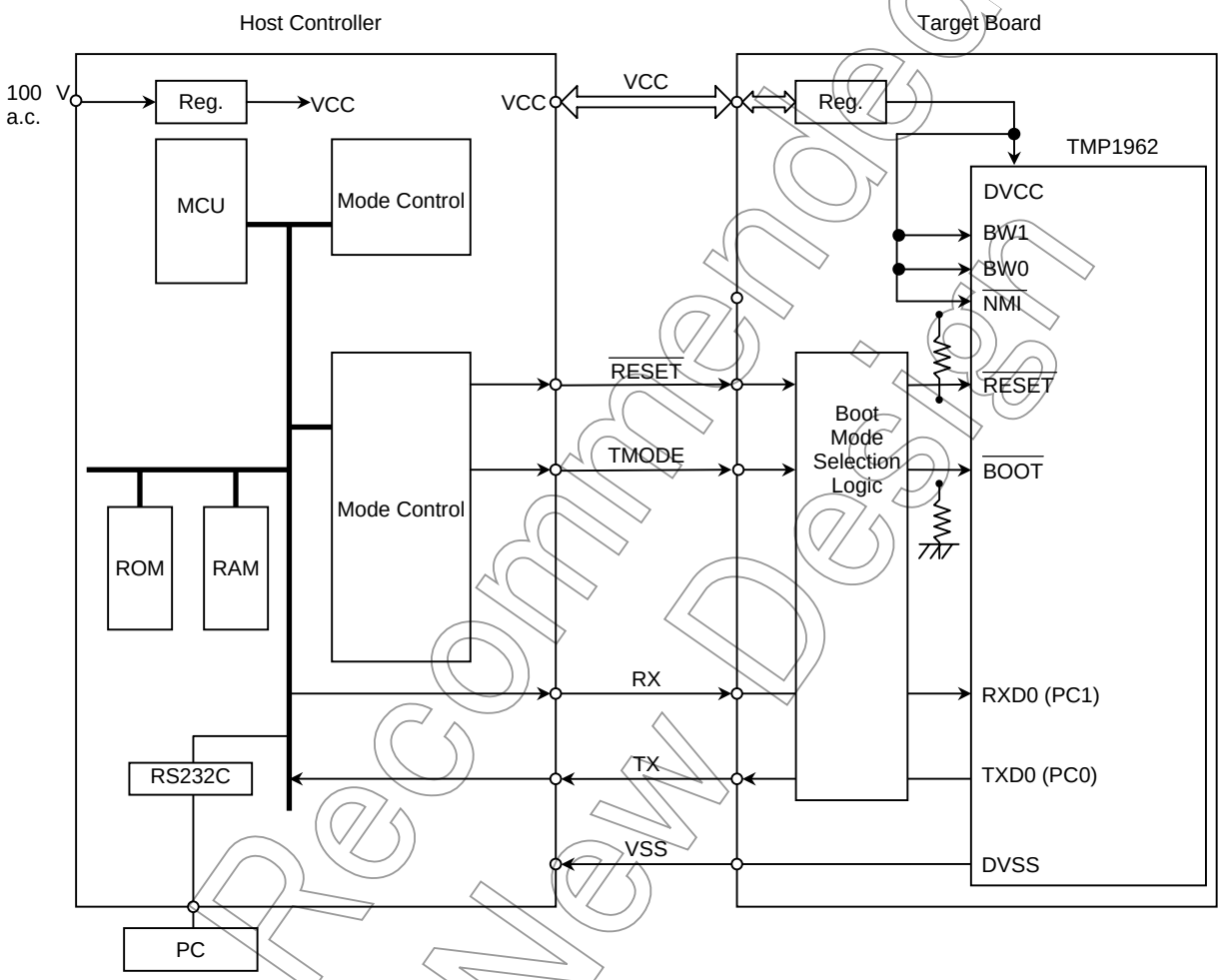



Figure 3.6 Example of a Connection Between a Host Controller and a Target Board (When the SIO0 is Configured for UART Mode)

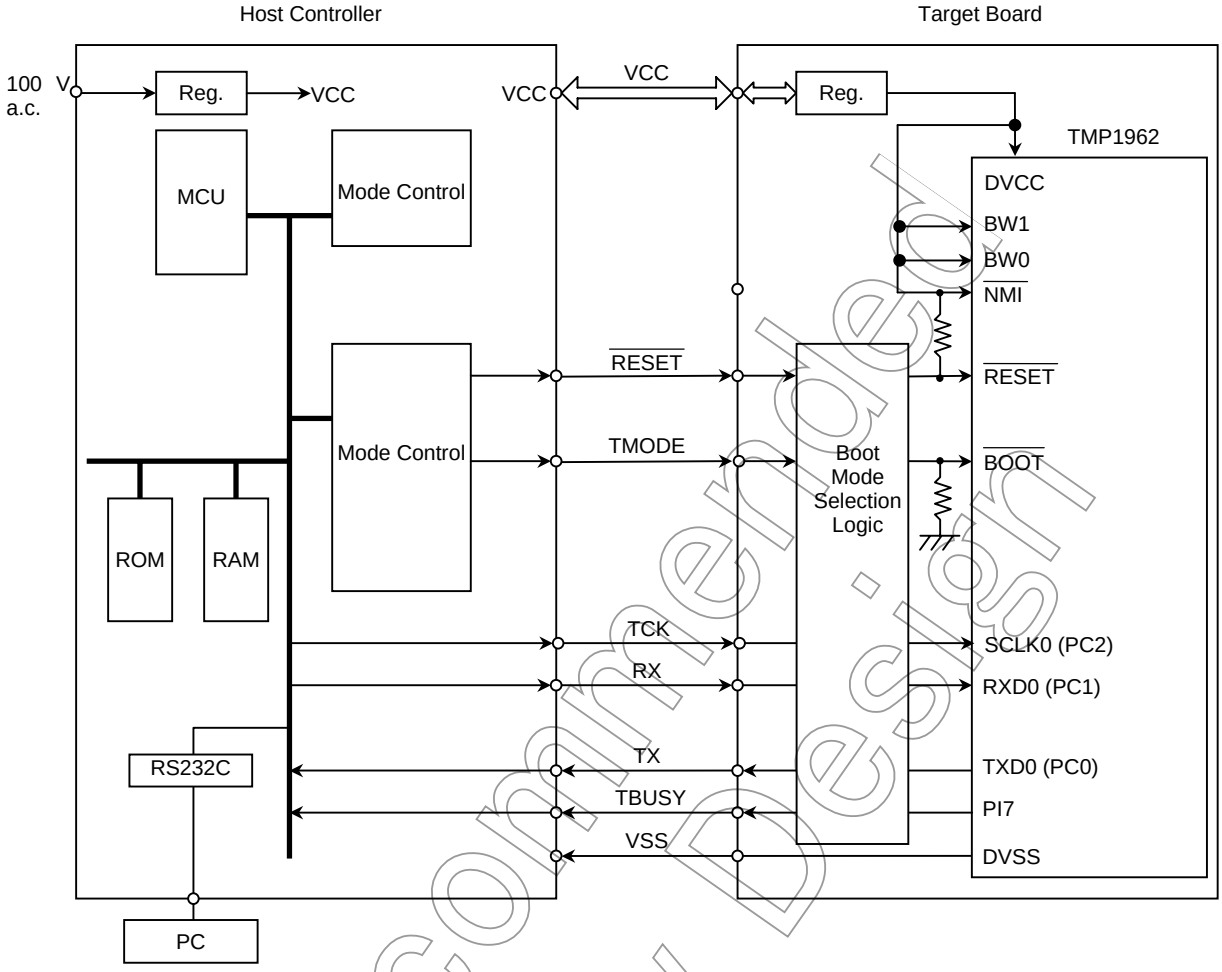



Figure 3.7 Example of a Connection Between a Host Controller and a Target Board (When the SIO0 is Configured for I/O Interface Mode)

### 3.5.2 Configuring for Single Boot Mode

For on-board programming, boot the TMP1962F10AXBG in Single Boot mode, as follows:

BW0 = 1

BW1 = 1

$\overline{\text{BOOT}}$ = 0

$\overline{\text{RESET}}$ = 0 → 1

Set the $\overline{\text{RESET}}$ input at logic 0, and the BW0, BW1 and $\overline{\text{BOOT}}$ (PJ2) inputs at the logic values shown above, and then release $\overline{\text{RESET}}$ (high).

### 3.5.3 Memory Map

Figure 3.8 shows a comparison of the memory maps in Normal and Single Boot modes. In single Boot mode, the on-chip flash memory is mapped to physical addresses (0x4000_0000 through 0x400F_FFFF), virtual addresses (0x0000_0000 through 0x000F_FFFF), and the on-chip boot ROM is mapped to physical addresses 0x1FC0_0000 through 0x1FC0_1FFF.

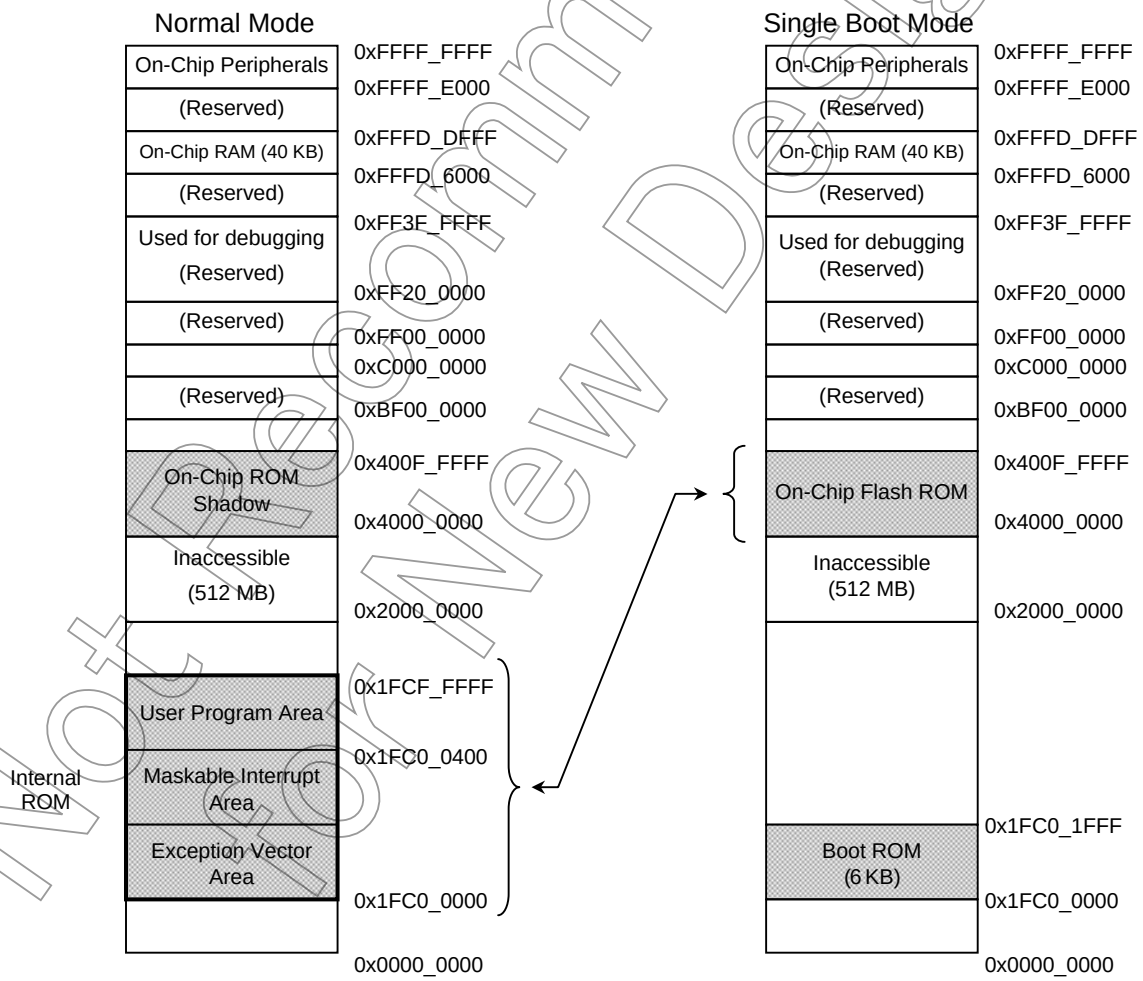



Figure 3.8 Memory Maps for Normal and Single Boot Modes (Physical Addresses)

### 3.5.4 Interface Specification

In Single Boot mode, an SIO channel is used for communications with a programming controller. Both UART (asynchronous) and I/O Interface (synchronous) modes are supported. The communication formats are shown below. In the subsections that follow, virtual addresses are indicated, unless otherwise noted.

- UART mode

  Communication channel: SIO Channel 0 (SIO0)
  Transfer mode: UART (asynchronous) mode, full-duplex
  Data length: 8 bits
  Parity bits: None
  STOP bits: 1
  Baud rate: Arbitrary baud rate

- I/O Interface mode

  Communication channel: SIO Channel 0 (SIO0)
  Transfer mode: I/O Interface mode, half-duplex
  Synchronization clock (SCLK0): Input
  Handshaking signal: PI7 configured as an output
  Baud rate: Arbitrary baud rate

Table 3.4 Required Pin Connections

| Pin | | Interface | |
|---|---|---|---|
| | | UART Mode | I/O Interface Mode |
| Power Supply Pins | DVCC2 (2.5 V) | Required | Required |
| | DVSS | Required | Required |
| Mode-Setting Pin | $\overline{BOOT}$ | Required | Required |
| Reset Pin | $\overline{RESET}$ | Required | Required |
| Communication Pins | TXD0 | Required | Required |
| | RXD0 | Required | Required |
| | SCLK0 | Not Required | Required (Input Mode) |
| | P87 | Not Required | Required (Input Mode) |

### 3.5.5 Data Transfer Format

The host controller is to issue one of the commands listed in Table 3.5 to the target board. Table 3.6 to Table 3.8 illustrate the sequence of two-way communications that should occur in response to each command.

Table 3.5 Single Boot Mode Commands

| Code | Command |
|---|---|
| 10H | RAM Transfer |
| 20H | Show Flash Memory Sum |
| 30H | Show Product Information |

Table 3.6 Transfer Format for the RAM Transfer Command

| | Byte | Data Transferred from the Controller to the TMP1962F10AXBG | Baud Rate | Data Transferred from the TMP1962F10AXBG to the Controller |
|---|---|---|---|---|
| Boot ROM | 1st byte | Serial operation mode and baud rate<br>For UART mode 86H<br>For I/O Interface mode 30H | Desired baud rate (Note 1) | — |
| | 2nd byte | — | | ACK for the serial operation mode byte<br>For UART mode<br>Normal acknowledge 86H<br>(The boot program aborts if the baud rate is can not be set correctly.)<br>For I/O Interface mode<br>Normal acknowledge 30H |
| | 3rd byte | Command code (10H) | | — |
| | 4th byte | — | | ACK for the command code byte (Note 2)<br>Normal acknowledge 10H<br>Negative acknowledge x1H<br>Communication error x8H |
| | 5th byte thru 16th byte | Password sequence (12 bytes)<br>(0x4000_03F4 thru 0x4000_03FF) | | — |
| | 17th byte | Checksum value for bytes 5–16 | | — |
| | 18th byte | — | | ACK for the checksum byte (Note 2)<br>Normal acknowledge 10H<br>Negative acknowledge 11H<br>Communication error 18H |
| | 19th byte | RAM storage start address (bits 31–24) | | — |
| | 20th byte | RAM storage start address (bits 23–16) | | — |
| | 21st byte | RAM storage start address (bits 15–8) | | — |
| | 22nd byte | RAM storage start address (bits 7–0) | | — |
| | 23rd byte | RAM storage byte count (bits 15–8) | | — |
| | 24th byte | RAM storage byte count (bits 7–0) | | — |
| | 25th byte | Checksum value for bytes 19–24 | | — |
| | 26th byte | — | | ACK for the checksum byte (Note 2)<br>Normal acknowledge 10H<br>Negative acknowledge 11H<br>Communication error 18H |
| | 27th byte thru mth byte | RAM storage data | | — |
| | (m + 1)th byte | Checksum value for bytes 27–m | | — |
| | (m + 2)th byte | — | | ACK for the checksum byte (Note 2)<br>Normal acknowledge 10H<br>Non-acknowledge 11H<br>Communications error 18H |
| RAM | (m + 3)th byte | — | | Jump to RAM storage start address |

**Note 1: In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.**

**Note 2: In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.**

**Note 3: The 19th to 25th bytes must be within the RAM address range 0xFFFD_6000–0xFFFF_DFFF**

**.**

Table 3.7 Transfer Format for the Show Flash Memory Sum Command

| | Byte | Data Transferred from the Controller to the TMP1962F10AXBG | Baud Rate | Data Transferred from the TMP1962F10AXBG to the Controller |
|---|---|---|---|---|
| Boot ROM | 1st byte | Serial operation mode and baud rate<br>For UART mode 86H<br>For I/O Interface mode 30H | Desired baud rate (Note 1) | — |
| | 2nd byte | — | | ACK for the serial operation mode byte<br>For UART mode<br>Normal acknowledge 86H<br>(The boot program aborts if the baud rate can not be set correctly.)<br>For I/O Interface mode<br>Normal acknowledge 30H |
| | 3rd byte | Command code (20H) | | — |
| | 4th byte | — | | ACK for the command code byte (Note 2)<br>Normal acknowledge 20H<br>Negative acknowledge x1H<br>Communication error x8H |
| | 5th byte | — | | SUM (upper byte) |
| | 6th byte | — | | SUM (lower byte) |
| | 7th byte | — | | Checksum value for bytes 5 and 6 |
| | 8th byte | (Wait for the next command code.) | | — |

**Note 1: In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.**

**Note 2: In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.**

Table 3.8 Transfer Format for the Show Product Information Command (1/2)

| | Byte | Data Transferred from the Controller to the TMP1962F10AXBG | Baud Rate | Data Transferred from the TMP1962F10AXBG to the Controller |
|---|---|---|---|---|
| Boot ROM | 1st byte | Serial operation mode and baud rate<br>For UART mode 86H<br>For I/O Interface mode 30H | Desired baud rate (Note 1) | — |
| | 2nd byte | — | | ACK for the serial operation mode byte<br>For UART mode<br>Normal acknowledge 86H<br>(The boot program aborts if the baud rate can not be set correctly.)<br>For I/O Interface mode<br>Normal acknowledge 30H |
| | 3rd byte | Command code (30H) | | — |
| | 4th byte | — | | ACK for the command code byte (Note 2)<br>Normal acknowledge 10H<br>Negative acknowledge x1H<br>Communication error x8H |
| | 5th byte | — | | Flash memory data<br>(at address 0x4000_03F0H) |
| | 6th byte | — | | Flash memory data<br>(at address 0x4000_03F1H) |
| | 7th byte | — | | Flash memory data<br>(at address 0x4000_03F2H) |
| | 8th byte | — | | Flash memory data<br>(at address 0x4000_03F3H) |
| | 9th byte<br>thru<br>20th byte | — | | Product name (12-byte ASCII code)<br>“TX1962F10” from the 9th byte |
| | 21st byte<br>thru<br>24th byte | — | | Password comparison start address (4 bytes)<br>F4H, 03H, 00H and 00H from the 21st byte |
| | 25th byte<br>thru<br>28th byte | — | | RAM start address (4 bytes)<br>00H, 60H, FDH and FFH from the 25th byte |
| | 29th byte<br>thru<br>32nd byte | — | | Dummy data (4 bytes)<br>FFH, 6FH, FDH and FFH from the 29th byte |
| | 33rd byte<br>thru<br>36th byte | — | | RAM end address (4 bytes)<br>FFH, DFH, FDH and FFH from the 33rd byte |
| | 37th byte<br>thru<br>40th byte | — | | Dummy data (4 bytes)<br>00H, 70H, FDH and FFH from the 37th byte |
| | 41st byte<br>thru<br>44th byte | — | | Dummy data (4 bytes)<br>FFH, EFH, FDH and FFH from the 41st byte |
| | 45th byte<br>thru<br>46th byte | — | | Fuse information (2 bytes)<br>01H and 00H from the 45th byte |
| | 47th byte<br>thru<br>50th byte | — | | Flash memory start address (4 bytes)<br>00H, 00H, 00H and 00H from the 47th byte |
| | 51st byte<br>thru<br>54th byte | — | | Flash memory end address (4 bytes)<br>FFH, FFH, 0FH and 00H from the 51st byte |
| | 55th byte<br>thru<br>56th byte | — | | Flash memory block count (2 bytes)<br>08H and 00H from at the 55th byte |

Table 3.8 Transfer Format for the Show Product Information Command (2/2)

| | Byte | Data Transferred from the Controller to the TMP1962F10AXBG | Baud Rate | Data Transferred from the TMP1962F10AXBG to the Controller |
|---|---|---|---|---|
| Boot ROM | 57th byte thru 60th byte | — | | Start address of a group of the same-size flash blocks (4 bytes) 00H, 00H, 00H and 00H from the 57th byte |
| | 61st byte thru 64th byte | — | | Size (in halfwords) of the same-size flash blocks (4 bytes) 00H, 00H, 01H and 00H from the 61st byte |
| | 65th byte | — | | Number of flash blocks of the same size (1 byte) 08H |
| | 66th byte | — | | Checksum value for bytes 5 to 65 |
| | 67th byte | (Wait for the next command code.) | | — |

**Note 1: In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.**

**Note 2: In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.**

### 3.5.6 Overview of the Boot Program Commands

When Single Boot mode is selected, the boot program is automatically executed on startup. The boot program offers these three commands, the details of which are provided on the following subsections.

- RAM Transfer command

  The RAM Transfer command stores program code transferred from a host controller to the on-chip RAM and executes the program once the transfer is successfully completed. The maximum program size is 36 kbytes. The RAM storage start address must be within the range.

  The RAM Transfer command can be used to download a flash programming routine of your own; this provides the ability to control on-board programming of the flash memory in a unique manner. The programming routine must utilize the flash memory command sequences described in Section 3.6.17

  Before initiating a transfer, the RAM Transfer command checks a password sequence coming from the controller against that stored in the flash memory. If they do not match, the RAM Transfer command aborts.

  Once the RAM Transfer command is complete, the whole on-chip RAM is accessible.

- Show Flash Memory Sum command

  The Show Flash Memory Sum command adds the contents of the 1024 kbytes of the flash memory together. The boot program does not provide a command to read out the contents of the flash memory. Instead, the Flash Memory Sum command can be used for software revision management.

- Show Product Information command

  The Show Product Information command provides the product name, on-chip memory configuration and the like. This command also reads out the contents of the flash memory locations at addresses 0x0000_03F0 through 0x0000_03F3. In addition to the Show Flash Memory Sum command, these locations can be used for software revision management.

### 3.5.7 RAM Transfer Command

See Table 3.6.

(1) The 1st byte specifies which one of the two serial operation modes is used. For a detailed description of how the serial operation mode is determined, see Section 3.5.11. If it is determined as UART mode, the boot program then checks if the SIO0 is programmable to the baud rate at which the 1st byte was transferred. During the first-byte interval, the RXE bit in the SC0MOD register is cleared.

- To communicate in UART mode

  Send, from the controller to the target board, 86H in UART data format at the desired baud rate. If the serial operation mode is determined as UART, then the boot program checks if the SIO0 can be programmed to the baud rate at which the first byte was transferred. If that baud rate is not possible, the boot program aborts, disabling any subsequent communications.

- To communicate in I/O Interface mode

  Send, from the controller to the target board, 30H in I/O Interface data format at 1/16 of the desired baud rate. Also send the 2nd byte at the same baud rate. Then send all subsequent bytes at a rate equal to the desired baud rate.

  In I/O Interface mode, the CPU sees the serial receive pin as if it were a general input port in monitoring its logic transitions. If the baud rate of the incoming data is high or the chip's operating frequency ishigh, the CPU may not be able to keep up with the speed of logic transitions. To prevent such situations, the 1st and 2nd bytes must be transferred at 1/16 of the desired baud rate; then the boot program calculates 16 times that as the desired baud rate.

  When the serial operation mode is determined as I/O Interface mode, the SIO0 is configured for SCLK Input mode. Beginning with the third byte, the controller must ensure that its AC timing restrictions are satisfied at the selected baud rate. In the case of I/O Interface mode, the boot program does not check the receive error flag; thus there is no such thing as error acknowledge (x8H).

(2) The 2nd byte, transmitted from the target board to the controller, is an acknowledge response to the 1st byte. The boot program echoes back the first byte: 86H for UART mode and 30H for I/O Interface mode.

- UART mode

  If the SIO0 can be programmed to the baud rate at which the 1st byte was transferred, the boot program programs the BR0CR and sends back 86H to the controller as an acknowledge. If the SIO0 is not programmable at that baud rate, the boot program simply aborts with no error indication.

  Following the 1st byte, the controller should allow for a time-out period of five seconds. If it does not receive 86H within the alloted time-out period, the controller should give up the communication.

  The boot program sets the RXE bit in the SC0MOD register to enable reception before loading the SIO transmit buffer with 86H.

- I/O Interface mode

  The boot program programs the SC0MOD0 and SC0CR registers to configure the SIO0 in I/O Interface mode (clocked by the rising edge of SCLK0), writes 30H to the SC0BUF. Then, the SIO0 waits for the SCLK0 signal to come from the controller. Following the transmission of the 1st byte, the controller should send the SCLK clock to the target board after a certain idle time (several microseconds). This must be done at 1/16 the desire baud rate. If the 2nd byte, which is from the target board to the controller, is 30H, then the controller should take it as a go-ahead. The controller must then delivers the 3rd byte to the target board at a rate equal to the desired baud rate. The boot program sets the RXE bit in the SC0MOD register to enable reception before loading the SIO transmit buffer with 30H.

(3) The 3rd byte, which the target board receives from the controller, is a command. The code for the RAM Transfer command is 10H.

(4) The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H and returns to the state in which it waits for a command again. In this case, the upper four bits of the acknowledge response are undefined — they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

If the 3rd byte is equal to any of the command codes listed in Table 3.5, the boot program echoes it back to the controller. When the RAM Transfer command was received, the boot program echoes back a value of 10H and then branches to the RAM Transfer routine. Once this branch is taken, a password check is done. Password checking is detailed in Section 3.5.12.

If the 3rd byte is not a valid command, the boot program sends back x1H to the controller and returns to the state in which it waits for a command again. In this case, the upper four bits of the acknowledge response are undefined — they hold the same values as the upper four bits of the previously issued command.

(5) The 5th to 16th bytes, which the target board receives from the controller, are a 12-byte password. The 5th byte is compared to the contents of address 0x0000_03F4 in the flash memory; the 6th byte is compared to the contents of address 0x0000_03F5 in the flash memory; likewise, the 16th byte is compared to the contents of address 0x0000_03FF in the flash memory. If the password checking fails, the RAM Transfer routine sets the password error flag.

(6) The 17th byte is a checksum value for the password sequence (5th to 16th bytes). To calculate the checksum value for the 12-byte password, add the 12 bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in Section 3.5.14.

(7) The 18th byte, transmitted from the target board to the controller, is an acknowledge response to the 5th to 17th bytes.

First, the RAM Transfer routine checks for a receive error in the 5th to 17th bytes. If there was a receive error, the boot program sends back 18H and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 5th to 17th bytes must result in zero (with the carry dropped). If it is not zero, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

Finally, the RAM Transfer routine examines the result of the password check. The following two cases are treated as a password error. In these cases, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

- Irrespective of the result of the password comparison, all of the 12 bytes of a password in the flash memory are the same value other than FFH.
- Not all of the password bytes transmitted from the controller matched those contained in the flash memory.

When all the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

(8) The 19th to 22nd bytes, which the target board receives from the controller, indicate the start address of the RAM region where subsequent data (e.g., a flash programming routine) should be stored. The 19th byte corresponds to bits 31–24 of the address, and the 22nd byte corresponds to bits 7–0 of the address.

(9) The 23rd and 24th bytes, which the target board receives from the controller, indicate the number of bytes that will be transferred from the controller to be stored in the RAM. The 23rd byte corresponds to bits 15–8 of the number of bytes to be transferred, and the 24th byte corresponds to bits 7–0 of the number of bytes.

(10) The 25th byte is a checksum value for the 19th to 24th bytes. To calculate the checksum value, add all these bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in Section 3.5.14.

(11) The 26th byte, transmitted from the target board to the controller, is an acknowledge response to the 19th to 25th bytes of data.

First, the RAM Transfer routine checks for a receive error in the 19th to 25th bytes. If there was a receive error, the RAM Transfer routine sends back 18H and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 19th to 25th bytes must result in zero (with the carry dropped). If it is not zero, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

- The RAM storage start address must be within the range 0xFFFD_6000–0xFFFD_EFFF.

When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

(12) The 27th to mth bytes from the controller are stored in the on-chip RAM of the TMP1962F10AXBG. Storage begins at the address specified by the 19th–22nd bytes and continues for the number of bytes specified by the 23rd–24th bytes.

(13) The (m+1)th byte is a checksum value. To calculate the checksum value, add the 27th to mth bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in Section 3.5.14.

(14) The (m+2)th byte is a acknowledge response to the 27th to (m+1)th bytes.

First, the RAM Transfer routine checks for a receive error in the 27th to (m+1)th bytes. If there was a receive error, the RAM Transfer routine sends back 18H and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

(15) Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 27th to (m+1)th bytes must result in zero (with the carry dropped). If it is not zero, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again. When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller. If the (m+2)th byte was a normal acknowledge response, a branch is made to the address specified by the 19th to 22nd bytes in 32-bit ISA mode.

### 3.5.8 Show Flash Memory Sum Command

See Table 3.7.

(1) The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

(2) The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Flash Memory Sum command is 20H.

(3) The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H and returns to the state in which it waits for a command again. In this case, the upper four bits of the acknowledge response are undefined — they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

If the 3rd byte is equal to any of the command codes listed in Table 3.5 on page 37, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 20H and then branches to the Show Flash Memory Sum routine.

If the 3rd byte is not a valid command, the boot program sends back x1H to the controller and returns

to the state in which it waits for a command again. In this case, the upper four bits of the acknowledge response are undefined — they hold the same values as the upper four bits of the previously issued command.

(4) The Show Flash Memory Sum routine adds all the bytes of the flash memory together. The 5th and 6th bytes, transmitted from the target board to the controller, indicate the upper and lower bytes of this total sum, respectively. For details on sum calculation, see Section 3.5.13.

(5) The 7th byte is a checksum value for the 5th and 6th bytes. To calculate the checksum value, add the 5th and 6th bytes together, drop the carry and take the two's complement of the sum. Transmit this checksum value from the controller to the target board.

(6) The 8th byte is the next command code.

### 3.5.9 Show Product Information Command

See Table 3.8.

(1) The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

(2) The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Product Information command is 30H.

(3) The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H and returns to the state in which it waits for a command again. In this case, the upper four bits of the acknowledge response are undefined — they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

If the 3rd byte is equal to any of the command codes listed in Table 3.5 on page 37, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 30H and then branches to the Show Flash Memory Sum routine.

If the 3rd byte is not a valid command, the boot program sends back x1H to the controller and returns to the state in which it waits for a command again. In this case, the upper four bits of the acknowledge response are undefined — they hold the same values as the upper four bits of the previously issued command.

(4) The 5th to 8th bytes, transmitted from the target board to the controller, are the data read from addresses 0x0000_03F0–0x0000_03F3 in the flash memory. Software version management is possible by storing a software id in these locations.

(5) The 9th to 20th bytes, transmitted from the target board to the controller, indicate the product name, which is "TX1962F10___" in ASCII code (where _ is a space).

(6) The 21st to 24th bytes, transmitted from the target board to the controller, indicate the start address of the flash memory area containing the password, i.e., F4H, 03H, 00H, 00H.

(7) The 25th to 28th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip RAM, i.e., 00H, 60H, FDH, FFH.

(8) The 29th to 32nd bytes, transmitted from the target board to the controller, are dummy data (FFH, 6FH, FDH, FFH).

(9) The 33rd to 36th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip RAM, i.e., FFH, FFH, FDH, FFH.

(10) The 37th to 40th bytes, transmitted from the target board to the controller, are 00H, 70H, FDH and FFH.

The 41st to 44th bytes, transmitted from the target board to the controller, are FFH, EFH, FDH and FFH.

(11) The 45th and 46th bytes, transmitted from the target board to the controller, indicate the presence or absence of the security and protect bits and whether the flash memory is divided into blocks. Bit 0 indicates the presence or absence of the security bit; it is 0 if the security bit is available. Bit 1 indicates the presence or absence of the protect bits; it is 0 if the protect bits are available. If bit 2 is 0, it indicates that the flash memory is divided into blocks. The remaining bits are undefined. The 45th and 46th bytes are 01H, 00H.

(12) The 47th to 50th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip flash memory, i.e., 00H, 00H, 00H, 00H.

(13) The 51st to 54th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip flash memory, i.e., FFH, FFH, 0FH, 00H.

(14) The 55th to 56th bytes, transmitted from the target board to the controller, indicate the number of flash blocks available, i.e., 08H, 00H.

(15) The 57th to 92nd bytes, transmitted from the target board to the controller, contain information about the flash blocks.

Flash blocks of the same size are treated as a group. Information about the flash blocks indicate the start address of a group, the size of the blocks in that group (in halfwords) and the number of the blocks in that group.

The 57th to 65th bytes are the information about the 128-kbyte blocks (Block 0 to Block 7). See Table 3.8 for the values of bytes transmitted.

(16) The 66th byte, transmitted from the target board to the controller, is a checksum value for the 5th to 65th bytes. The checksum value is calculated by adding all these bytes together, dropping the carry and taking the two’s complement of the total sum.

(17) The 67th byte is the next command code.

### 3.5.10 Acknowledge Responses

The boot program represents processing states with specific codes. Table 3.9 to Table 3.11 show the values of possible acknowledge responses to the received data. The upper four bits of the acknowledge response are equal to those of the command being executed. Bit 3 of the code indicates a receive error. Bit 0 indicates an invalid command error, a checksum error or a password error. Bit 1 and bit 2 are always 0. Receive error checking is not done in I/O Interface mode.

Table 3.9 ACK Response to the Serial Operation Mode Byte

| Return Value | Meaning |
|---|---|
| 86H | The SIO can be configured to operate in UART mode. (See Note) |
| 30H | The SIO can be configured to operate in I/O Interface mode. |

**Note: If the serial operation mode is determined as UART, the boot program checks if the SIO can be programmed to the baud rate at which the operation mode byte was transferred. If that baud rate is not possible, the boot program aborts, without sending back any response.**

Table 3.10 ACK Response to the Command Byte

| Return Value | Meaning |
|---|---|
| x8H (See Note) | A receive error occurred while getting a command code. |
| x1H (See Note) | An undefined command code was received. (Reception was completed normally.) |
| 10H | The RAM Transfer command was received. |
| 20H | The Show Flash Memory Sum command was received. |
| 30H | The Show Product Information command was received. |

**Note: The upper four bits of the ACK response are the same as those of the previous command code.**

Table 3.11 ACK Response to the Checksum Byte

| Return Value | Meaning |
|---|---|
| 18H | A receive error occurred. |
| 11H | A checksum or password error occurred. |
| 10H | The checksum was correct. |

### 3.5.11 Determination of a Serial Operation Mode

The first byte from the controller determines the serial operation mode. To use UART mode for communications between the controller and the target board, the controller must first send a value of 86H at a desired baud rate to the target board. To use I/O Interface mode, the controller must send a value of 30H at 1/16 the desired baud rate. Figure 3.9 shows the waveforms for the first byte.

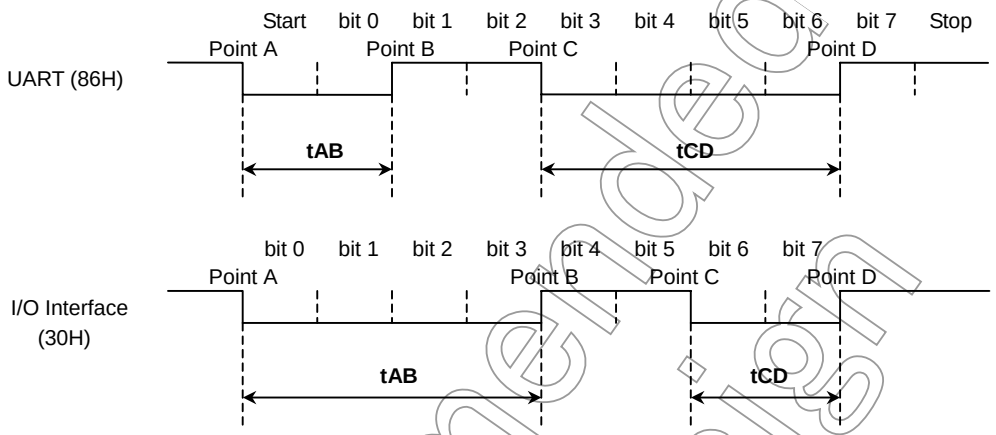



Figure 3.9 Serial Operation Mode Byte

After $\overline{\text{RESET}}$ is released, the boot program monitors the first serial byte from the controller, with the SIO reception disabled, and calculates the intervals of tAB, tAC and tAD. Figure 3.10 shows a flowchart describing the steps to determine the intervals of tAB, tAC and tAD. As shown in the flowchart, the boot program captures timer counts each time a logic transition occurs in the first serial byte. Consequently, the calculated tAB, tAC and tAD intervals are bound to have slight errors. If the transfer goes at a high baud rate, the CPU might not be able to keep up with the speed of logic transitions at the serial receive pin. In particular, I/O Interface mode is more prone to this problem since its baud rate is generally much higher than that for UART mode. To avoid such a situation, the controller should send the first serial byte at 1/16 the desired baud rate.

The flowchart in Figure 3.11 shows how the boot program distinguishes between UART and I/O Interface modes. If the length of tAB is equal to or less than the length of tCD, the serial operation mode is determined as UART mode. If the legnth of tAB is greater than the length of tCD, the serial operation mode is determined as I/O Interface mode. Bear in mind that if the baud rate is too high or the timer operating frequency is too low, the timer resolution will be coarse, relative to the intervals between logic transitions. This becomes a problem due to inherent errors caused by the way in which timer counts are captured by software; consequently the boot program might not be able to determine the serial operation mode correctly.

For example, the serial operation mode may be determined to be I/O Interface mode when the intended mode is UART mode. To avoid such a situation, when UART mode is utilized, the controller should allow for a time-out period within which it expects to receive an echo-back (86H) from the target board. The controller should give up the communication if it fails to get that echo-back within the alloted time. When I/O Interface mode is utilized, once the first serial byte has been transmitted, the controller should send the SCLK clock after a certain idle time to get an acknowledge response. If the received acknowledge response is not 30H, the controller should give up further communications.

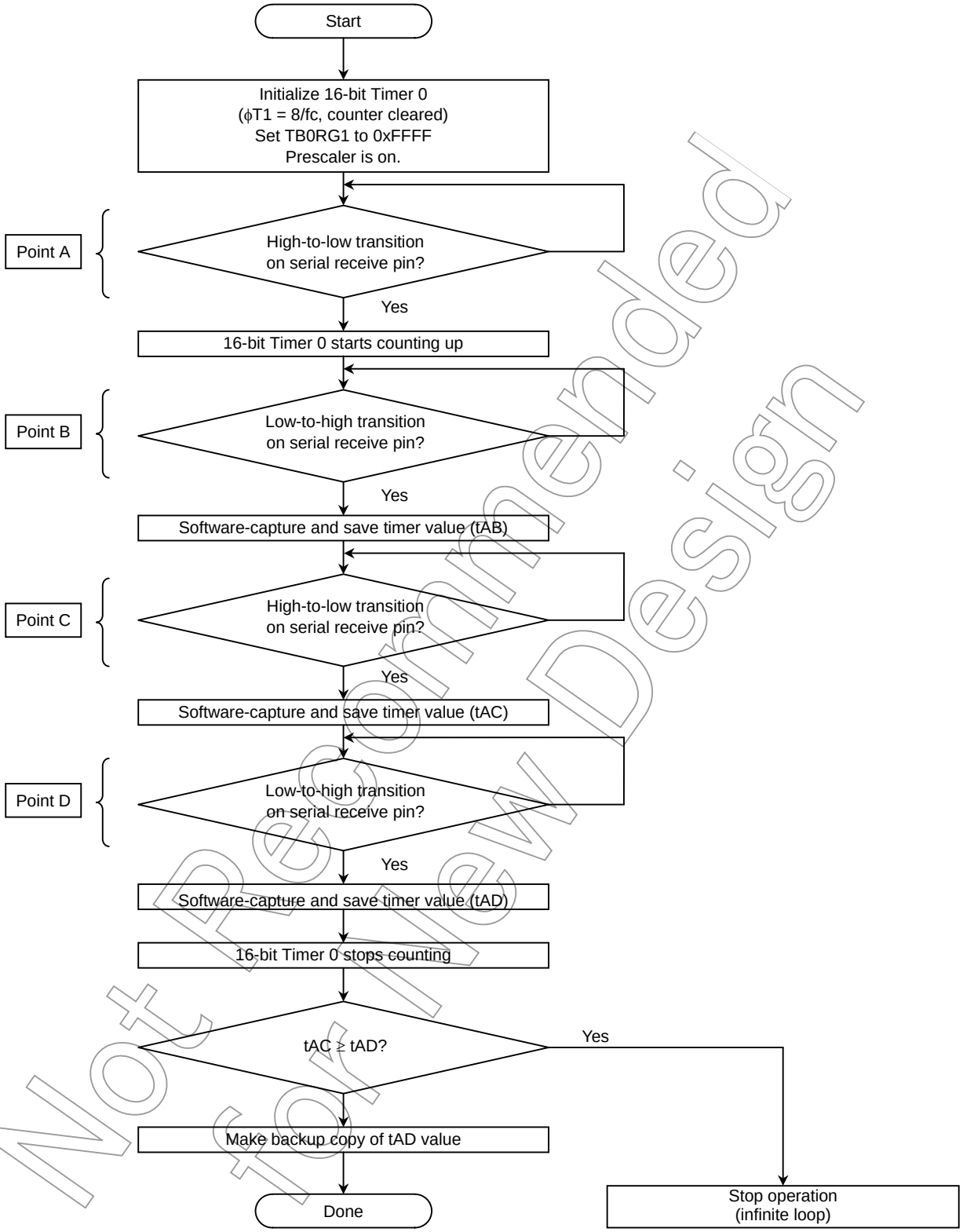





Figure 3.10 Serial Operation Mode Byte Reception Flow

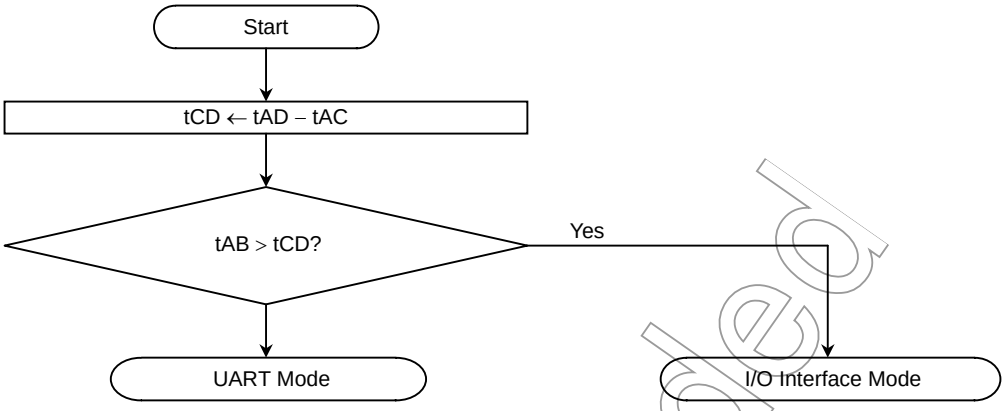



Figure 3.11 Serial Operation Mode Determination Flow

### 3.5.12 Password

The RAM Transfer command (10H) causes the boot program to perform a password check. Following an echo-back of the command code, the boot program checks the contents of the 12-byte password area (0x4000_03F4 to 0x4000_03FF) within the flash memory. If all these address locations contain the same bytes of data other than FFH, a password area error occurs. In this case, the boot program returns an error acknowledge (11H) in response to the checksum byte (the 17th byte), regardless of whether the password sequence sent from the controller is all FFHs.

The password sequence received from the controller (5th to 16th bytes) is compared to the password stored in the flash memory. Table 3.12 shows how they are compared byte-by-byte. All of the 12 bytes must match to pass the password check. Otherwise, a password error occurs, which causes the boot program to return an error acknowledge in response to the checksum byte (the 17th byte).

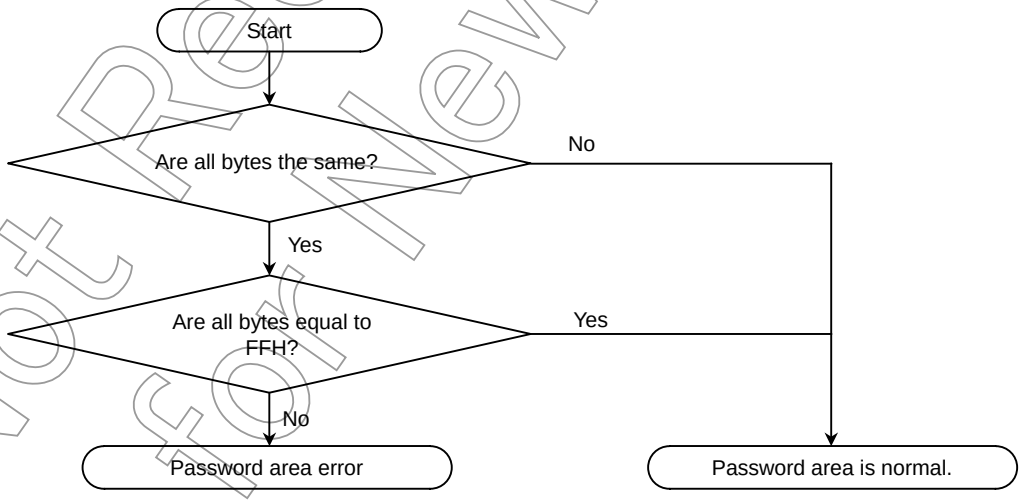



Figure 3.12 Password Area Check Flow

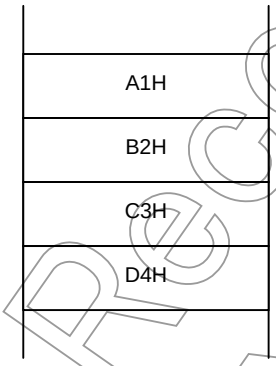
Table 3.12 Relationship between Received Bytes and Flash Memory Locations

| Received Byte | Compared Flash Memory Data |
|---|---|
| 5th byte | Address 0x0000_03F4 |
| 6th byte | Address 0x0000_03F5 |
| 7th byte | Address 0x0000_03F6 |
| 8th byte | Address 0x0000_03F7 |
| 9th byte | Address 0x0000_03F8 |
| 10th byte | Address 0x0000_03F9 |
| 11th byte | Address 0x0000_03FA |
| 12th byte | Address 0x0000_03FB |
| 13th byte | Address 0x0000_03FC |
| 14th byte | Address 0x0000_03FD |
| 15th byte | Address 0x0000_03FE |
| 16th byte | Address 0x0000_03FF |

### 3.5.13 Calculation of the Show Flash Memory Sum Command

The Show Flash Memory Sum command adds all 1024 kbytes of the flash memory together and provides the total sum as a halfword quantity. The sum is sent to the controller, with the upper eight bits first, followed by the lower eight bits.

Example:




For the interest of simplicity, assume the depth of the flash memory is four locations. Then the sum of the four bytes is calculated as:

A1H + B2H + C3H + D4H = 02EAH

Hence, 02H is first sent to the controller, followed by EAH.

### 3.5.14 Checksum Calculation

The checksum byte for a series of bytes of data is calculated by adding the bytes together, dropping the carries, and taking the two's complement of the total sum. The Show Flash Memory Sum command and the Show Product Information command perform the checksum calculation. The controller must perform the same checksum operation in transmitting checksum bytes.

Example:

Assume the Show Flash Memory Sum command provides the upper and lower bytes of the sum as E5H and F6H. To calculate the checksum for a series of E5H and F6H:

(1) Add the bytes together.

E5H + F6H = 1DBH

(2) Drop the carry.

(3) Take the two's complement of the sum, and that is the checksum byte.

0 – DBH = 25H

### 3.5.15 General Boot Program Flowchart

Figure 3.13 shows an overall flowchart of the boot program.

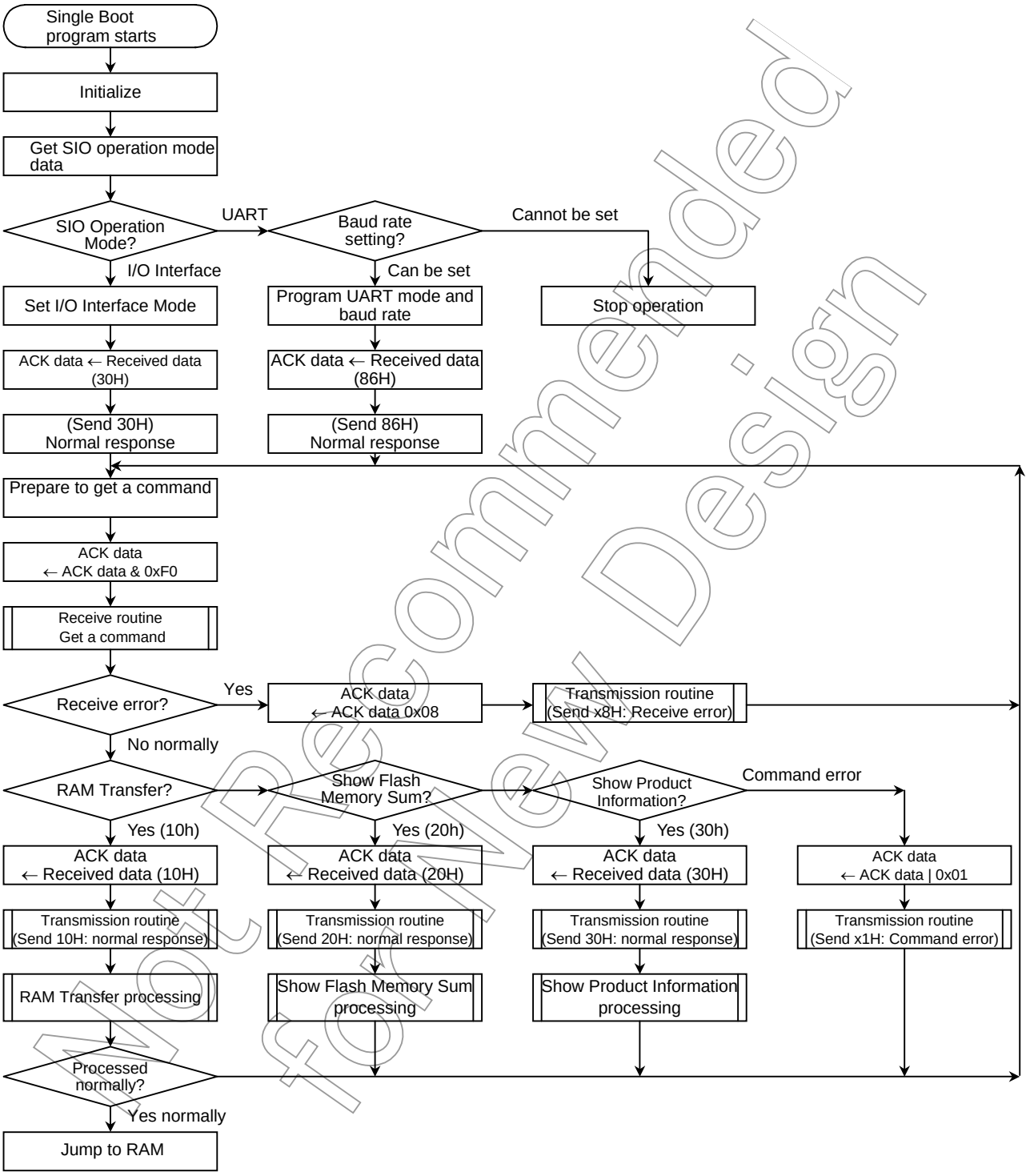





Figure 3.13 Overall Boot Program Flow

## 3.6 On-Board Programming and Erasure

The TMP1962F10AXBG flash memory is command set compatible with the JEDEC EEPROM standard, with a few exceptions. In User Boot mode and Single Boot mode (the RAM Transfer command), the flash memory can be programmed and erased by the CPU executing software commands. It is the user's responsibility to create a program/erase routine. Because the flash memory can not be read while it is being programmed or erased, the program/erase routine must be executed out of the on-chip RAM or an external memory device.

### 3.6.1 Key Features

The TMP1962F10AXBG flash memory commands are in principle compatible with the standard JEDEC commands. For program/erase operations, the system can issue a command sequence to the flash memory by using CPU instructions such as LD. After the command sequence is written, the flash memory does not require the system to provide further controls or timings. The flash memory initiates the embedded program or erase algorithm automatically. The entire flash memory or one or more flash blocks can be erased at a time.

Table 3.14 Flash Memory Features

| Feature | Description |
|---|---|
| Auto Program | Programs and verifies the desired addresses word by word automatically. |
| Auto Chip Erase | Erases and verifies the entire memory array automatically. |
| Auto Block Erase | Erases and verifies all memory locations in the selected block automatically. |
| Auto Multi-Block Erase | Erases and verifies all memory locations in multiple selected blocks automatically. |
| Write operation status | Provides several status bits such as the Data Polling bit and Toggle bit, which can be used to determine whether a program or erase operation is complete or in progress. |
| Security feature | Prevents intrusive access to the flash memory while in Programmer mode. When the security feature is turned off, the entire memory array is erased and verified automatically, regardless of whether a given block is protected or not. |
| Block protection | Disables both program and erase operations in any block. |

Bear in mind that, due to the on-chip CPU interface, the TMP1962F10AXBG uses addresses different from those of the standard flash command sequences. Unless otherwise noted, programming is done word by word; thus the word load instruction should be used to write to the flash array.

### 3.6.2 Block Architecture

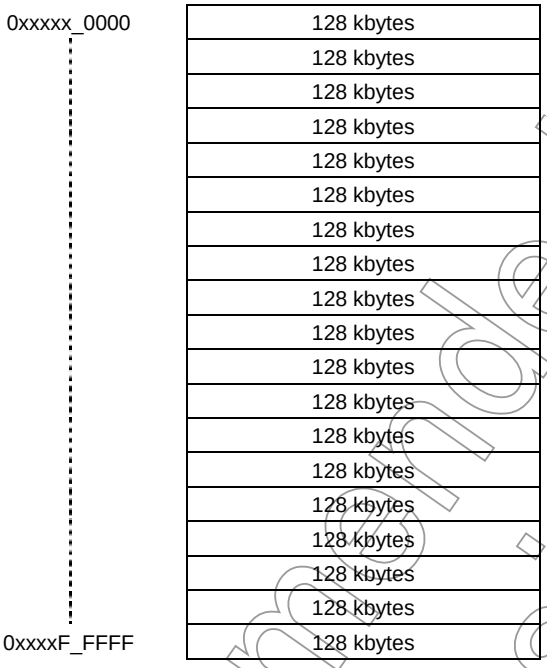



Figure 3.16 Flash Memory Block Architecture

### 3.6.3 CPU-to-Flash Interface

Figure 3.17 illustrates the internal interface between the CPU and the flash memory in on-board programming modes. The diagram does not show the actual logic network; instead it is only a conceptual depiction of the CPU-to-flash interface.

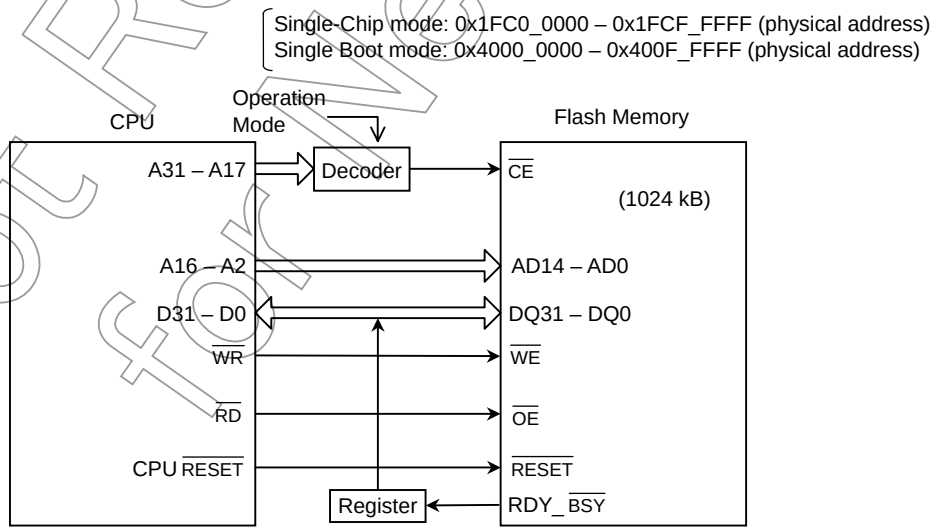



Figure 3.17 Internal CPU-to-Flash Interface

### 3.6.4 Read Mode and Embedded Operation Mode

The flash memory of the TMP1962F10AXBG has the following two modes of operation:

- Read mode in which array data is read
- Embedded Operation mode in which the flash array is programmed or erased

The flash memory enters Embedded Operation mode when a valid command sequence is executed in Read mode. In Embedded Operation mode, array data can not be read.

### 3.6.5 Reading Array Data

The flash memory is automatically set to reading array data upon CPU reset after device power-up and after an embedded operation is successfully completed. If an embedded operation terminated abnormally or the flash memory is required to return to the Read mode, the Read/Reset command (software reset) or hardware reset is used.

### 3.6.6 Writing Commands

The operations of the flash memory are selected by commands or command sequences written into the internal command register. This uses the same mechanism as for JEDEC-standard EEPROMs. Commands are made up of data sequences written at specific addresses via the command register. See Table 3.16 on page 63 for the list of command sequences.

The command sequence being written can be canceled by issuing the Read/Reset command between sequence cycles. The Read/Reset command clears the command register and resets the flash memory to Read mode. Invalid command sequences also cause the flash memory to clear the command register and return to Read mode.

### 3.6.7 Reset

- Read/Reset command (software reset)

  The flash memory does not return to Read mode if an embedded operation terminated abnormally. In this case, the Read/Reset command must be issued to put the flash memory back in Read mode. The Read/Reset command may also be written between sequence cycles of the command being written to clear the command register.

- Hardware reset ( $\overline{\text{RESET}}$ input)

  As shown in Figure 3.17, the flash memory has a reset pin, which is connected to the reset signal of the CPU. When the system drives the $\overline{\text{RESET}}$ pin to $V_{IL}$ or when certain events such as a watchdog timer time-out causes a CPU reset, the flash memory immediately terminates any operation in progress and is reset to Read mode.

  The Read/Reset command is also tied to the $\overline{\text{RESET}}$ pin to reset the flash memory to Read mode. The embedded operation that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence because data may be corrupted.

  For a description of the hardware reset operation, see Section 3.3.2, *Reset Operation*. When a valid reset is achieved, the CPU reads the Reset exception vector from the flash memory and services the Reset exception.

### 3.6.8 Auto Program Command

A bit must be programmed to change its state from a 1 to a 0. A bit can not be programmed from a 0 back to a 1. Only an erase operation can change a 0 back to a 1.

In User Boot mode and the RAM Transfer command of Single Boot mode, the Auto Program command programs the desired addresses word by word. The Auto Program command requires four bus cycles; the program address and data are written in the fourth cycle, upon completion of which the program operation will commence. As programming is performed on a word-by-word basis, the program address must be a multiple of four.

Writing data shorter than a 32-bit word requires special considerations for the bits that are not to be altered. The word in the memory does not need to be in the erased state prior to programming. If the word is in the erased state, a 32-bit write must be performed, with all the bits not to be altered set to 1.

Examples:

- When a word location is in the erased state

  To program the least-significant byte of that word to 55H, 0xFFFF_FF55 must be written to the word address.

**Note : The superscription of data cannot be done in the flash memory.**

The Auto Program command executes a sequence of internally timed events to program the desired bits of the addressed memory word and verify that the desired bits are sufficiently programmed. The system can determine the status of the programming operation by using write status flags (see Table 3.19 on page 65).

Any commands written during the programming operation are ignored. A hardware reset immediately terminates the programming operation. The programming operation that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence because data may be corrupted.

The block protection feature disables programming operations in any block. If an attempt is made to program a protected block, the Auto Program command does nothing; the flash memory returns to Read mode in approximately 3 μm after the completion of the fourth bus cycle of the command sequence.

When the embedded Auto Program algorithm is complete, the flash memory returns to Read mode.

If any failure occurs during the programming operation, the flash memory remains locked in Embedded Operation mode. The system can determine this status by using write status flags. To put the flash memory back in Read mode, use the Read/Reset command to reset the flash memory or a hardware reset to reset the whole chip. In case of a programming failure, it is recommended to replace the chip or discontinue the use of the failing flash block.

### 3.6.9 Auto Chip Erase Command

The Auto Chip Erase command requires six bus cycles. The flash area is partitioned into two areas, that are Block 0 to Block 3 (Flash 0) and Block 4 to Block 7 (Flash 1). The chip erase operation is performed for individual area. Set A[19] (address 19) = 0 for Flash 0, and A[19] = 1 for Flash 1 by each bus cycle. After completion of the sixth bus cycle, the Auto Chip Erase operation will commence immediately. The embedded Auto Chip Erase algorithm automatically preprograms the entire memory for an all-0 data pattern prior to the erase; then it automatically erases and verifies the entire memory for an all-1 data pattern. The system can determine the status of the chip erase operation by using write status flags (see

Table 3.19 on page 65).

Any commands written during the chip erase operation are ignored. A hardware reset immediately terminates the chip erase operation. The chip erase operation that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence because data may be corrupted.

The block protection feature disables erase operations in any block. The Auto Chip Erase algorithm erases the unprotected blocks and ignores the protected blocks. If all the blocks are protected, the Auto Chip Erase command does nothing; the flash memory returns to Read mode in approximately 100 μm after the completion of the sixth bus cycle of the command sequence.

When the embedded Auto Chip Erase algorithm is complete, the flash memory returns to Read mode.

If any failure occurs during the erase operation, the flash memory remains locked in Embedded Operation mode. The system can determine this status by using write status flags. To put the flash memory back in Read mode, use the Read/Reset command to reset the flash memory or a hardware reset to reset the whole chip. In case of an erase failure, it is recommended to replace the chip or discontinue the use of the failing flash block. The failing block can be identified by means of the Block Erase command.

### 3.6.10 Auto Block Erase and Auto Multi-Block Erase Commands

The Auto Block Erase command requires six bus cycles. A time-out begins from the completion of the command sequence. After a time-out, the erase operation will commence. The embedded Auto Block Erase algorithm automatically preprograms the selected block for an all-0 data pattern, and then erases and verifies that block for an all-1 data pattern.

During the time-out period, additional block addresses and Auto Block Erase commands may be written. Multi-block is selectable in either Flash 0 area or Flash 1 area.

Any command other than Auto Block Erase during the time-out period resets the flash memory to Read mode. The block erase time-out period is 50 μm. The system may read DQ3 to determine whether the time-out period has expired. The block erase timer begins counting upon completion of the sixth bus cycle of the Auto Block Erase command sequence. The system can determine the status of the erase operation by using write status flags (see Table 3.19 on page 65).

Any commands written during the block erase operation are ignored. A hardware reset immediately terminates the block erase operation. The block erase operation that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence because data may be corrupted.

The block protection feature disables erase operations in any block. The Auto Block Erase algorithm erases the unprotected blocks and ignores the protected blocks. If all the selected blocks are protected, the Auto Block Erase algorithm does nothing; the flash memory returns to Read mode in approximately 100 μm after the final bus cycle of the command sequence. When the embedded Auto Block Erase algorithm is complete, the flash memory returns to Read mode.

If any failure occurs during the erase operation, the flash memory remains locked in Embedded Operation mode. The system can determine this status by using write status flags. To put the flash memory back in Read mode, use the Read/Reset command to reset the flash memory or a hardware reset to reset the whole chip. In case of an erase failure, it is recommended to replace the chip or discontinue the use of the failing flash block. If any failure occurred during the multi-block erase operation, the failing block can be identified by running Auto Block Erase on each of the blocks selected for multi-block erasure.

### 3.6.11 Block Protect Command

The block protection feature disables both program and erase operations in any block. After completion of the seventh bus write cycle, FCLS<bit2> is 0 during Block Protect operation, and 1 after Block Protect operation.

Table 3.15 Effects of the Program and Erase Commands on the Protected Blocks

| Command | Operation |
|---|---|
| Program command on a protected block | No programming operation is performed, and the flash memory automatically returns to Read mode. |
| Block Erase command on a protected block | No erase operation is performed, and the flash memory automatically returns to Read mode. |
| Chip Erase command when all the blocks are protected | No erase operation is performed, and the flash memory automatically returns to Read mode. |
| Chip Erase command when any blocks are protected | Only the unprotected blocks are erased. Upon completion, the flash memory automatically returns to Read mode. |
| Multi-Block Erase command when any blocks are protected | Only the unprotected blocks are erased. Upon completion, the flash memory automatically returns to Read mode. |

Any commands written during the Block Protect algorithm are ignored. A hardware reset immediately terminates the block protect operation. The Block Protect command that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence.

### 3.6.12 Block Unprotect Operation

Block unprotect operation is performed for individual area (Flash 0 and Flash 1). Set A[19] in accordance to the area required for block unprotect by the bus cycle. After completion of the seventh bus write cycle, FCLS<bit2> is 0 during block unprotect operation, and 1 after block unprotect operation.

Any commands written during the Block Unprotect algorithm are ignored. The hardware reset immediately terminates the block unprotect operation. The Block Protect command should be reinitiated once the flash memory is ready to accept another command sequence. Use the Verify Block Protect command to verify the protect status of a block.

### 3.6.13 Verify Block Protect Command

The Verify Block Protect command is used to verify the protect status of a block. Verify Block Protect is a four-bus-cycle operation. The address of the block to be verified is given in the fourth cycle. Any address within the block range will suffice, provided A0 = A1 = A2 = A3 = 0, A4 = 1 and A6 = 0. To get correct data, a 32-bit read must be performed. Use the last read as valid data. If the selected block is protected, a value of 0x0000_0001 is returned. If the selected block is not protected, a value of 0x0000_0000 is returned. Following the fourth bus cycle, an additional block address may be read.

The Verify Block Protect command does not return the flash memory to Read mode. Either the Read/Reset command or a hardware reset is required to reset the flash memory to Read mode or to write the next command.

### 3.6.14 Write Operation Status

As shown in Table 3.19, the flash memory provides several flag bits to determine the status of an embedded operation: DQ7, DQ5 and DQ3. These status bits can be read during an embedded operation using the same timing as for Read mode. The flash memory automatically returns to Read mode when an embedded operation completes. The status of an embedded program operation can be monitored to determine whether the hardware sequence flag during an embedded operation, or the data read after completion of an embedded operation match the cell data. Read of the hardware sequence flag is performed by checking start of embedded operation (FLCS<bit2>=0).

During the embedded program operation, the system must provide the program address (with A0 = 0 and A1 = 0) to read valid status information. During the embedded erase operation, the system must provide an address (with A0 = 0 and A1 = 0) within any of the blocks selected for erasure to read valid status information.

- DQ7 (Data Polling)

  The Data Polling bit, DQ7, indicates to the host system the status of the embedded operation. Data Polling is valid after the final bus write cycle of an embedded command sequence.

  When the embedded Program algorithm is in progress, an attempt to read the flash memory will produce the complement of the data last written to DQ7. Upon completion of the embedded Program algorithm, an attempt to read the flash memory will produce the true data last written to DQ7. Therefore, the system can use DQ7 to determine whether the embedded Program algorithm is in progress or complete.

  When the embedded Erase algorithm is in progress, an attempt to read the flash memory will produce a 0 at the DQ7 output. Upon completion of the embedded Erase algorithm, the flash memory will produce a 1 at the DQ7 output.

  If there is a failure during an embedded operation, DQ7 continues to output the same value. Thus, DQ7 must always be polled in conjunction with the Exceeded Timing Limits (DQ5) flag. Figure 3.21 shows the DQ7 polling algorithm.

  The flash memory disables address latching when an embedded operation is complete. Data polling must be performed with a valid programmed address or an address within any of the non-protected blocks selected for erasure.

- DQ5 (Exceeded Timing Limits)

  DQ5 produces a 0 while the program or erase operation is in progress normally. DQ5 produces a 1 to indicate that the program or erase time has exceeded the specified internal limit. This is a failure condition that indicates the program or erase cycle was not successfully completed.

  The DQ5 failure condition also appears if the system tries to program a 1 to a location that was previously programmed to a 0. Only an erase operation can change a 1 back to a0. In this case, the embedded Program algorithm halts the operation. Once the operation has exceeded the timing limits, DQ5 will indicate a 1. Note that this is not a device failure condition since the flash memory was used incorrectly.

  Under both these conditions, the flash memory remains locked in Embedded Operation mode. The system must issue the Read/Reset command to return the flash memory to Read mode.

- DQ3 (Block Erase Timer)

  After the completion of the sixth bus cycle of the Auto Block Erase command sequence, the block erase time-out window of 50 μm begins. The erase operation will begin after the time-out has expired. When the time-out is complete and the erase operation has begun, DQ3 switches from 0 to 1. If DQ3 is 0, the flash memory will accept additional Auto Block Erase commands. Each time an Auto Block Erase command is written, the time-out window is reset. To ensure that the command has been accepted, the system should check DQ3 prior to and following each Auto Block Erase command. If DQ3 is 1 on the second status check, the command might not have been accepted.

### 3.6.15 Flash Control/Status Register

This is an 8-bit register that indicates the Ready/Busy status of an embedded algorithm and controls the security feature.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| FLCS | Bit Symbol | FLRMSK | | | | | RDY/BSY | | |
| (0xFFFF_E520) | Read/Write | W | | | | | R | | |
| | Reset Value | 0 | | | | | 1 | 0 | |
| | Function | Flash reset mask enable<br>1: Reset a flash control circuitry<br>0: Unreset a flash control circuitry | | | | | Ready/ Busy<br>0: Embedded algorithm is in progress.<br>1: Embedded algorithm is complete. | Must be written as “0”. | |

Figure 3.18 Flash Control/Status Register

- Bit 2: Ready/Busy Flag Bit(

  In Programmer mode, the ALE pin functions as the RDY/$\overline{\text{BSY}}$ pin. The host system can monitor the state of this pin to determine whether an embedded algorithm is in progress or complete. The CPU can poll the RDY/BSY bit in the FLCS register for the same purpose. The RDY/BSY bit is cleared to 0 when the flash memory is actively erasing or programming. The RDY/BSY bit is set to 1 when an embedded operation has completed and the flash memory is ready to accept the next command. If any failure occurs during the program or erase operation, this bit remains cleared. A hardware reset sets this bit.

  The RDY/BSY bit is cleared upon completion of the final bus write cycle of an embedded operation command, with one exception. In the case of the Auto Block Erase command, this bit is cleared after the time-out has expired. Any command is ignored while the RDY/BSY bit is cleared.

- Bit 7: Flash Reset Mask Bit

  An interval of 30 μs (T.B.D.) is allowed to elapse before starting the processor core operation after a reset upon power on, which is required to initialize an on-chip flash control circuitry. SYSRDY is signaled to indicate the start of the processor core operation from outside of TMP1962. After the processor core is reset, SYSRDY switches from Low to High. The reset operation after power on (do not power off) is controlled by bit 7 (FLRMSK) of the flash control/status register (FLCS) in the flash control part. When the FLRMSK bit is 0 (initial value), the flash control circuitry is always initialized. When the FLRMSK bit is set to 1, the flash control circuitry is not initialized. (Read of the on-chip flash memory is performed correctly.) In this case, the interval of 30 μs (T.B.D.) is not required for the flash control circuitry to be stable after a reset. Immediately the processor core starts operation, and the SYSRDY signal switches to High. The value set to the FLRMSK bit is retained until power off. Set FLRMSK=1 commonly by initial setting afterreset.

**Note: The Flash Control/Status register must be accessed as a 32-bit quantity.**

### 3.6.16 Flash Security

The TMP1962F10AXBG flash memory supports not only on-board programming but also programming using a general-purpose programmer. Therefore, the TMP1962F10AXBG flash memory provides a security feature to prevent intrusive access to the flash memory while in Programmer mode.

The TMP1962F10AXBG is secured by all eight blocks being protected, and the contents of a flash memory can not be read by a programmer.

- Securing the flash (Disabling read accesses)

  Securing the flash memory disables a general-purpose programmer to read its contents. To turn on the security feature, once programming is complete, protect all eight blocks. That secures the flash memory. If one of eight blocks is unprotected, the flash memory is unsecured.

  In on-board operating modes, the CPU can read the flash memory even if the security is on. When the security is ON, any reads by programming equipment will always return a word-length value of 0x0098.

- Unsecuring the flash (Enabling read accesses)

  The security feature is designed to disable reads of the flash memory by programming equipment. While the TMP1962F10AXBG is soldered on a board, the CPU can always read the flashmemory, regardless of whether or not the security is on. Since the flash memory is placed under control of a user's application program in on-board operating modes, it is not easy for third parties to perform intrusive access to the flash memory. Therefore, within the confines of a board, the flash memory does not need to be secured. To turn off the security feature, unprotect eight blocks. Unsecuring the flash memory enables the flash memory erase operation to occur before turning off the security feature. After a flash memory has been erased, the flash memory unsecure operation is completed by erasing the Block Protect bit.

### 3.6.17 Command Definition

Table 3.16 On-Board Programming Mode Command Definition

| Command Sequence | Cycles Required | Bus Cycles | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 1st Cycle (Write) | | 2nd Cycle (Write) | | 3rd Cycle (Write) | | 4th Cycle (Read/Write) | | 5th Cycle (Read/Write) | |
| | | Addr | Data | Addr | Data | Addr | Data | Addr | Data | Addr | Data |
| Read/Reset | 1 | 0xXXXX | F0H | | | | | | | | |
| Read/Reset | 3 | 0x5554 | AAH | 0xAAA8 | 55H | 0x5554 | F0H | RA | RD | | |
| Auto Program | 4 | 0x5554 | AAH | 0xAAA8 | 55H | 0x5554 | A0H | PA | PD | | |
| Auto Chip Erase | 6 | 0x5554 | AAH | 0xAAA8 | 55H | 0x5554 | 80H | 0x5554 | AAH | 0xAAA8 | 55H |
| Auto Block Erase | 6 | 0x5554 | AAH | 0xAAA8 | 55H | 0x5554 | 80H | 0x5554 | AAH | 0xAAA8 | 55H |
| Block Protect | 7 | 0x5554 | AAH | 0xAAA8 | 55H | 0x5554 | 9AH | 0x5554 | AAH | 0xAAA8 | 55H |
| Block Unprotect | 7 | 0x5554 | AAH | 0xAAA8 | 55H | 0x5554 | 6AH | 0x5554 | AAH | 0xAAA8 | 55H |
| ID Read/Block Protect Verify | 4 | 0x5554 | AAH | 0xAAA8 | 55H | 0x5554 | 90H | IA/BPA | ID/BD | | |

(Continued from above)

| Command Sequence | Cycles Required | Bus Cycles | | | |
|---|---|---|---|---|---|
| | | 6th Cycle (Write) | | 7th Cycle (Write) | |
| | | Addr | Data | Addr | Data |
| Read/Reset | 1 | | | | |
| Read/Reset | 3 | | | | |
| Auto Program | 4 | | | | |
| Auto Chip Erase | 6 | 0x5554 | 10H | | |
| Auto Block Erase | 6 | BA | 30H | | |
| Block Protect | 7 | 0x5554 | 9AH | BPA | 9AH |
| Block Unprotect | 7 | 0x5554 | 6AH | 0x5554 | 6AH |
| Auto Security On (Note 1) | 4 | | | | |

**Note 1: After every bus write cycle, execute SYNC and NOP in sequence.**
**Note 2: Set the value corresponding the flash memory address to 16-bit through 19-bit in every bus write cycle.**

The addresses to be provided by the CPU are shown below.

Table 3.17 Addresses Provided by the CPU

| Command Address | CPU Addresses: A23–A0 | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A23–A16 | A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
| 0xXXX0 | Flash memory block | X | X | X | X | X | X | X | X | X | X | X | X | 0 | 0 | 0 | 0 |
| 0x0000 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0xAAA8 | | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 0x5554 | | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |

- F0H, AAH, 55H, A0H, 80H, 10H, 30H:

  Command data. Write command data as a byte quantity.

- RA: Read Address

  RD: Read Data

- PA: Program Address

  PD: Program Data

  The address must be a multiple of four. Write data on a word-by-word basis.

- BA: Block Address (BA0–BA6)

  Refer to Table 3.18.

- BPA: Verify Block Protect Address

  BD: Block Protect Data

  Refer to Table 3.18. The address of the block to be verified can be any of the addresses within the block, with A6 = 0, A4 = 1, A3 = 0, A1 = 0 and A0 = 0. If a block is protected, a value of 0x0000_0001 will be returned. If a block is not protected, a value of 0x0000_0000 will be returned.

- IA: ID Read Address
- ID: ID Data

Table 3.18 Block Erase Addresses

| | | User Boot Mode | Boot Mode | | A19 | A18 | A17 |
|---|---|---|---|---|---|---|---|
| Flash0 | Block-0 | 0x1FC0_0000 - 0x1FC1_FFFF (or 0x4000_0000 - 0x4001_FFFF) | 0x1FC0_0000 - 0x1FC1_FFFF | 128 kbyte | 0 | 0 | 0 |
| | Block-1 | 0x1FC2_0000 - 0x1FC3_FFFF (or 0x4002_0000 - 0x4003_FFFF) | 0x1FC2_0000 - 0x1FC3_FFFF | 128 kbyte | 0 | 0 | 1 |
| | Block-2 | 0x1FC4_0000 - 0x1FC5_FFFF (or 0x4004_0000 - 0x4005_FFFF) | 0x1FC4_0000 - 0x1FC5_FFFF | 128 kbyte | 0 | 1 | 0 |
| | Block-3 | 0x1FC6_0000 - 0x1FC7_FFFF (or 0x4006_0000 - 0x4007_FFFF) | 0x1FC6_0000 - 0x1FC7_FFFF | 128 kbyte | 0 | 1 | 1 |
| Flash1 | Block-4 | 0x1FC8_0000 - 0x1FC9_FFFF (or 0x4008_0000 - 0x4009_FFFF) | 0x1FC8_0000 - 0x1FC9_FFFF | 128 kbyte | 1 | 0 | 0 |
| | Block-5 | 0x1FCA_0000 - 0x1FCB_FFFF (or 0x400A_0000 - 0x400B_FFFF) | 0x1FCA_0000 - 0x1FCB_FFFF | 128 kbyte | 1 | 0 | 1 |
| | Block-6 | 0x1FCC_0000 - 0x1FCD_FFFF (or 0x400C_0000 - 0x400D_FFFF) | 0x1FCC_0000 - 0x1FCD_FFFF | 128 kbyte | 1 | 1 | 0 |
| | Block-7 | 0x1FCE_0000 - 0x1FCF_FFFF (or 0x400E_0000 - 0x400E_FFFF) | 0x1FCE_0000 - 0x1FCF_FFFF | 128 kbyte | 1 | 1 | 1 |

The address of the block to be erased can be any of the addresses within that block with A0=0 and A1=0.

Example: To select BA0 in User Boot mode, provide any address in the range between 0x1FC0_0000 and 0x1FC1_FFFF.

Table 3.19 Write Status Flags

| Status | | D7 (DQ7) | D5 (DQ5) | D3 (DQ3) |
|---|---|---|---|---|
| Embedded operation in progress | Auto Program | $\overline{DQ7}$ | 0 | 0 |
| | Auto Erase (during the time-out window) | 0 | 0 | 0 |
| | Auto Erase | 0 | 0 | 1 |
| Time-out in embedded operation | Auto Program | $\overline{DQ7}$ | 1 | 1 |
| | Auto Erase | 0 | 1 | 1 |

**Note: D31–D8, D6, D4 and D2–D0 are don't-cares.**

### 3.6.18 Embedded Algorithms

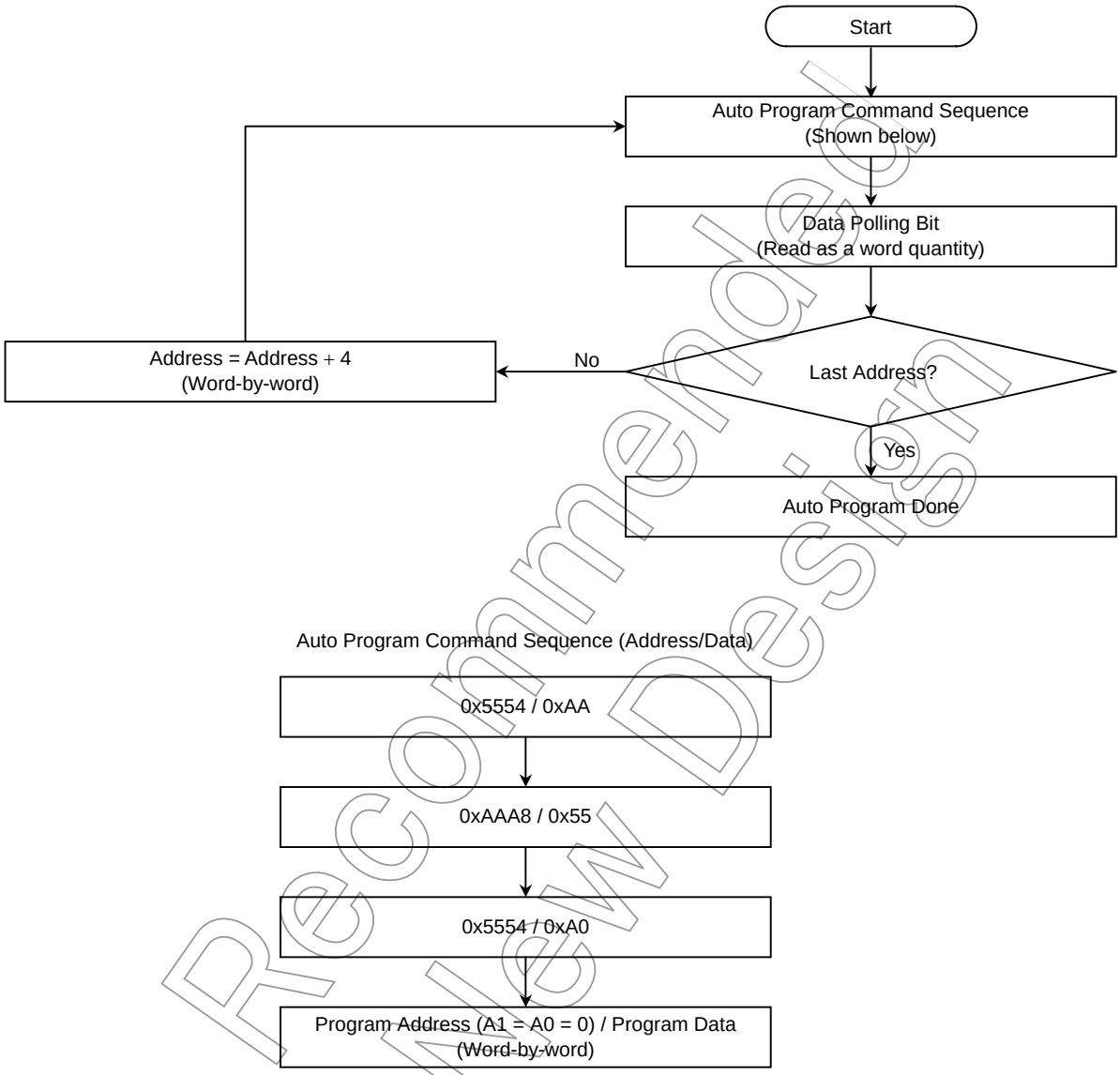



Figure 3.19 Auto Program Operation


Figure 3.20 Auto Erase Operations

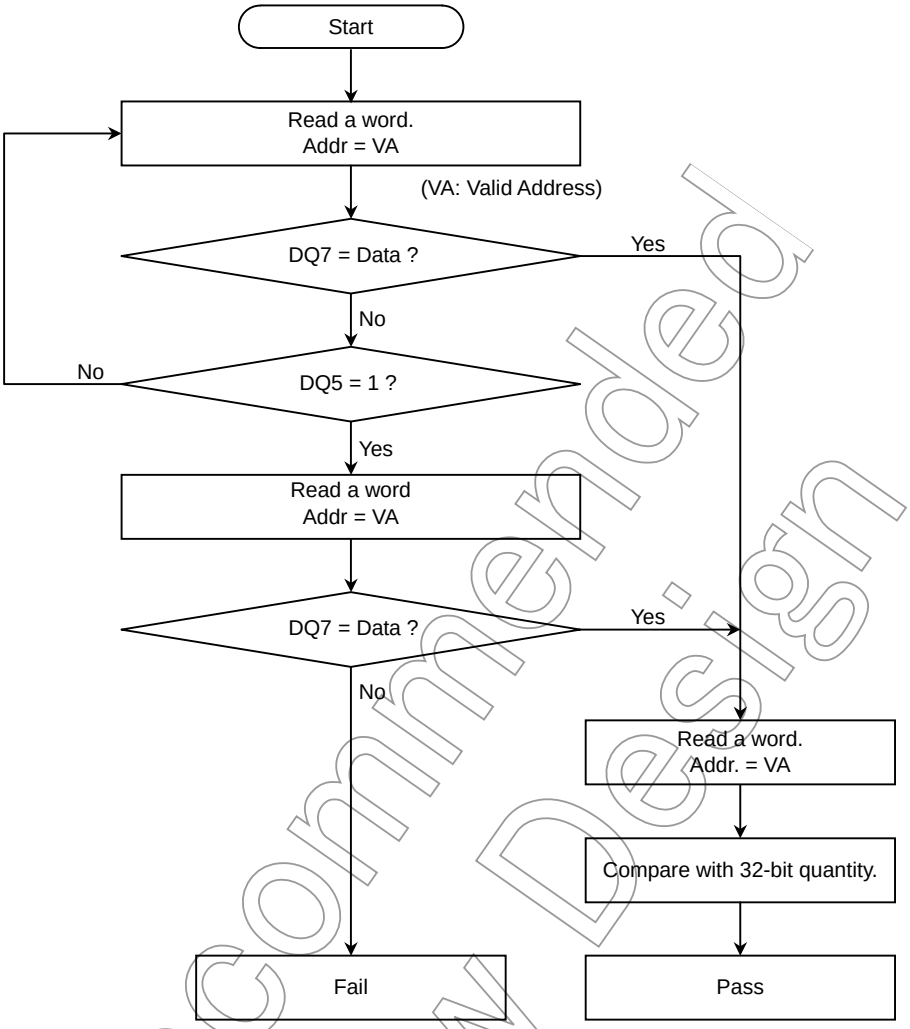



Figure 3.21 Data Polling (DQ7) Algorithm

# 4. Electrical Characteristics

The letter x in equations presented in this chapter represents the cycle period of the fsys clock selected through the programming of the SYSCR1.SYSCK bit. The fsys clock may be derived from either the high-speed or low-speed crystal oscillator. The programming of the clock gear function also affects the fsys frequency. All relevant values in this chapter are calculated with the high-speed (fc) system clock (SYSCR1.SYSCK = 0) and a clock gear factor of 1/fc (SYSCR1.GEAR[1:0] = 00).

## 4.1 Absolute Maximum Ratings

| Parameter | | Symbol | Rating | Unit |
|---|---|---|---|---|
| Supply voltage | | $V_{CC2}$ (Core) | −0.3 to 3.6 | V |
| | | $V_{CC3}$ (I/O) | −0.3 to 4.0 | |
| | | AVCC (A/D) | −0.3 to 3.6 | |
| | | FVCC3 (L1 Pin) | −0.3 to 4.0 | |
| Input voltage | | $V_{IN}$ | −0.3 to $V_{CC}$ + 0.3 | V |
| Low-level output current | Per pin | $I_{OL}$ | 5 | mA |
| | Total | $\Sigma I_{OL}$ | 50 | |
| High-level output current | Per pin | $I_{OH}$ | −5 | |
| | Total | $\Sigma I_{OH}$ | 50 | |
| Power dissipation (Ta = 85°C) | | PD | 600 | mW |
| Soldering temperature (10 s) | | $T_{SOLDER}$ | 260 | °C |
| Storage temperature | | $T_{STG}$ | −65 to 150 | °C |
| Operating temperature | Except during flash W/E | $T_{OPR}$ | −20 to 85 | °C |
| | During flash W/E | | 10 to 60 | |
| Write/erase cycles | | $N_{EW}$ | 100 | cycle |

$V_{CC}2$ = DVCC21 = DVCC22 = FVCC2 = CVCC2, $V_{CC}3$ = DVCC3n (n = 1 to 4),
AVCC = AVCC31 = AVCC32, $V_{SS}$ = DVSS = FVSS = AVSS = CVSS

**Note: Absolute Maximum Ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions. The equipment manufacturer should design so that no Absolute Maximum Ratings value is exceeded with respect to current, voltage, power dissipation, temperature, etc. Exposure to conditions beyond those listed above may cause permanent damage to the device or affect device reliability, which could increase potential risks of personal injury due to IC blowup and/or burning.**

## 4.2 DC Electrical Characteristics (1/4)

Ta = –20 to 85°C

| Parameter | | Symbol | Conditions | Min | Typ (Note 1) | Max | Unit |
|---|---|---|---|---|---|---|---|
| Supply voltage<br>FVCC2 = CVCC2<br>= DVCC21 = DVCC22<br>FVSS = CVSS = DVSS = 0 V | | DVCC2m<br>(m = 1 to 2) | fosc = 10 to 13.5 MHz<br>fsys = 3.75 to 40.5 MHz<br>PLLON, INTLV = “H” | 2.2 | | 2.7 | V |
| | | DVCC3n<br>(n = 1 to 4) | fsys = 3.75 to 40.5 MHz | 1.65 | | 3.3 | |
| | | FVCC3 | fsys = 3.75 to 40.5 MHz | 2.9 | | 3.6 | |
| Low-level input voltage | P7-P9<br>(Used as a port) | $V_{IL1}$ | 2.7 V ≤ AVCC32 ≤ AVCC31 ≤ 3.5 V | –0.3 | | 0.3 AVCC31 | V |
| | | | 1.65 ≤ AVCC32 ≤ < 2.7 V | | | 0.3 AVCC32 | |
| | P0-P6,<br>PA-PC, PD0-PD6,<br>PE0-PE2, PF2-PF7,<br>PG-PH, PI7,<br>PJ1-PJ4, PL-PP | $V_{IL2}$ | 1.65 V ≤ DVCC3n ≤ 3.3 V (n = 1 to 4) | | | 0.3 DVCC3n | |
| | | | 2.2 V ≤ DVCC2m ≤ 2.7 V (m = 1 to 2) | | | 0.2 DVCC2m | |
| | PD7, PE3-PE7,<br>PF0-PF1, PI0-PI6,<br>PJ0, PK, $\overline{\text{PLLOFF}}$ ,<br>RSTPUP, $\overline{\text{RESET}}$<br>$\overline{\text{DRESET}}$ , $\overline{\text{DBGE}}$<br>SDI/ $\overline{\text{DINT}}$ , TCK, TMS,<br>TDI, $\overline{\text{TRST}}$<br>$\overline{\text{NMI}}$ , BW0, BW1 | $V_{IL3}$ | 2.7 V ≤ DVCC3n ≤ 3.3 V (n = 1 to 4) | | | 0.15 DVCC3n | |
| | | | 1.65 V ≤ DVCC3n < 2.7 V (n = 1 to 4)<br>2.2 V ≤ DVCC2m ≤ 2.7 V (m = 1 to 2) | | | 0.1 DVCC3n<br>0.1 DVCC2m | |
| | X1 | $V_{IL4}$ | 2.2 V ≤ CVCC2 ≤ 2.7 V | | | 0.1 CVCC2 | |

Note 1: Ta = 25°C, DVCC3n = 3.0 V, DVCC2m = 2.5 V, AVCC3 = 3.3 V, unless otherwise noted.

## 4.3 DC Electrical Characteristics (2/4)

Ta = –20 to 85°C

| Parameter | | Symbol | Conditions | | Min | Typ. (Note 1) | Max | Unit |
|---|---|---|---|---|---|---|---|---|
| High-level input voltage | P7-P9 (Used as a port) | $V_{IH1}$ | 2.7 V ≤ AVCC32 ≤ AVCC31 ≤ 3.5 V | | 0.7 AVCC31 | | | V |
| | | | 1.65 ≤ AVCC32 < 2.7 V | | 0.7 AVCC32 | | | |
| | P0-P6, PA-PC, PD0-PD6, PE0-PE2, PF2-PF7, PG-PH, PI7, PJ1-PJ4, PL-PP | $V_{IH2}$ | 1.65 V ≤ DVCC3n ≤ 3.3 V (n = 1 to 4) | | 0.7 DVCC3n | | | |
| | | | 2.2 V ≤ DVCC2m ≤ 2.7 V (m = 1 to 2) | | 0.8 DVCC2m | | | |
| | PD7, PE3-PE7, PF0-PF1, PI0-PI6, PJ0, PK, $\overline{\text{PLLOFF}}$, RSTPUP, $\overline{\text{RESET}}$, $\overline{\text{DRESET}}$, $\overline{\text{DBGE}}$, SDI/$\overline{\text{DINT}}$, TCK, TMS, TDI, $\overline{\text{TRST}}$, $\overline{\text{NMI}}$, BW0, BW1 | $V_{IH3}$ | 2.7 V ≤ DVCC3n ≤ 3.3 V (n = 1 to 4) | | 0.85 DVCC3n | | DVCC2m + 0.3 DVCC3n + 0.3 | |
| | | | 1.65 V ≤ DVCC3n < 2.7 V (n = 1 to 4)<br>2.2 V ≤ DVCC2m ≤ 2.7 V (m = 1 to 2) | | 0.9 DVCC3n<br>0.9 DVCC2m | | | |
| | X1 | $V_{IH4}$ | 2.2 V ≤ CVCC2 ≤ 2.7 V | | 0.9 CVCC2 | | | |
| Low-level output voltage | | $V_{OL}$ | $I_{OL}$ = 2 mA | DVCC3n ≥ 2.7 V | | | 0.4 | V |
| | | | $I_{OL}$ = 500 μA | DVCC3n < 2.7 V<br>DVCC2m ≤ 2.7 V | | | 0.2 DVCC3n ≤ 0.4<br>0.2 DVCC2 ≤ 0.4 | |
| High-level output voltage | | $V_{OH}$ | $I_{OH}$ = –2 mA | DVCC3n ≥ 2.7 V | 2.4 | | | |
| | | | $I_{OH}$ = –500 μA | DVCC3n < 2.7 V<br>DVCC2m ≤ 2.7 V | 0.8 DVCC3n<br>0.8 DVCC2 | | | |

Note 1: Ta = 25°C, DVCC3n = 3.0 V, DVCC2m = 2.5 V, AVCC3 = 3.3 V, unless otherwise noted.

## 4.4 DC Electrical Characteristics (3/4)

Ta = –20 to 85°C

| Parameter | Symbol | Conditions | Min | Typ. (Note 1) | Max | Unit |
|---|---|---|---|---|---|---|
| Input leakage current | $I_{LI}$ | $0.0 \le V_{IN} \le$ DVCC2m (m = 1 to 2)<br>$0.0 \le V_{IN} \le$ DVCC3n (n = 1 to 4)<br>$0.0 \le V_{IN} \le$ AVCC31<br>$0.0 \le V_{IN} \le$ AVCC32 | | 0.02 | ±5 | μA |
| Output leakage current | $I_{LO}$ | $0.2 \le V_{IN} \le$ DVCC2m – 0.2 (m = 1 to 2)<br>$0.2 \le V_{IN} \le$ DVCC3n – 0.2 (n = 1 to 4)<br>$0.2 \le V_{IN} \le$ AVCC31 – 0.2<br>$0.2 \le V_{IN} \le$ AVCC32 – 0.2 | | 0.05 | ±10 | |
| Power-down voltage (STOP mode RAM backup) | $V_{STOP}$ (DVCC2) | $V_{IL2}$ = 0.2DVCC2m, $V_{IL3}$ = 0.1DVCC2m<br>$V_{IH2}$ = 0.8DVCC2m, $V_{IH3}$ = 0.9DVCC2m | 2.2 | | 2.7 | V |
| Pull-up resister at Reset | RRST | 2.2 V ≤ DVCC21 ≤ 2.7 V | 20 | 50 | 240 | kΩ |
| Schmitt W Idth<br>PD7, PE3-PE7, PF0-PF1, PI0-PI6, PJ0, PK, $\overline{\text{PLLOFF}}$, RSTPUP, $\overline{\text{RESET}}$, $\overline{\text{DRESET}}$, $\overline{\text{DBGE}}$, SDI/$\overline{\text{DINT}}$, TCK, TMS, TDI, $\overline{\text{TRST}}$, $\overline{\text{NMI}}$, BW0, BW1 | VTH | 2.7 V ≤ DVCC3n ≤ 3.3 V (n = 2, 4) | 0.4 | 0.9 | | V |
| | | 1.65 V ≤ DVCC3n < 2.7 V (n = 2, 4)<br>2.2 V ≤ DVCC2m ≤ 2.7 V (m = 1 to 2) | 0.3 | 0.6 | | |
| Programmable pull-up/ pull-down resistor<br>P32-P37,P40-P43<br>KEY0-KEYD, DRESET, $\overline{\text{DBGE}}$, SDI/$\overline{\text{DINT}}$, TCK, TMS, TDI, $\overline{\text{TRST}}$ | PKH | DVCC3n = 3.0 V ±0.3 V (n = 2 to 4) | 15 | 50 | 100 | kΩ |
| | | DVCC3n = 2.5 V ±0.2 V (n = 2 to 4) | 20 | 50 | 240 | |
| | | DVCC3n = 2.0 V ±0.2 V (n = 2 to 4) | 25 | 160 | 600 | |
| Pin capacitance (Except power supply pins) | $C_{IO}$ | fc = 1 MHz | | | 10 | pF |

Note 1: Ta = 25°C, DVCC3n = 3.0 V, DVCC2m = 2.5 V, AVCC3 = 3.3 V, unless otherwise noted.

## 4.5 DC Electrical Characteristics (4/4)

DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ± 0.3 V,
DVCC3n = 3.0 V ± 0.3 V, AVCC3m = 3.3 V ± 0.2 V

Ta = -20 to 85°C (n = 1 to 4, m = 1, 2)

| Parameter | Symbol | Conditions | Min | Typ. (Note 1) | Max | Unit |
|---|---|---|---|---|---|---|
| NORMAL (Note 2): Gear = 1/1 | $I_{CC}$ | $f_{sys}$ = 40.5 MHz ($f_{OSC}$ = 13.5 MHz, PLLON) INTLV = "H" | | 90 | 110 | mA |
| IDLE (Doze) | | | | 40 | 60 | |
| IDLE (Halt) | | | | 33 | 50 | |
| STOP | | DVCC2m = FVCC2 = CVCC2 = 2.2 to 2.7 V<br>DVCC3n = 1.65 to 3.3 V<br>AVCC3m = 2.7 to 3.5 V<br>FVCC3 = 3.0 to 3.6 V | | 55 | 900 | μA |

Note 1: Ta = 25°C, DVCC2m = 2.5 V, DVCC3n = 3.0 V, AVCC3m = 3.3 V, unless otherwise noted.

Note 2: Measured with the CPU dhrystone operating, all I/O peripherals channel on, and 16-bit external bus operated with 4 system clocks.

Note 3: The supply current flowing through the DVCC2m, FVCC2, DVCC3n, CVCC2, FVCC3 and AVCC3m pins is included in the digital supply current parameter (ICC).

## 4.6 ADC Electrical Characteristics

DVCC15=CVCC15=1.5V±0.15V,DVCC2＝2.5V±0.2V, DVCC3n＝ 3.0V±0.3V ,
AVCC3m＝3.0V±0.2V **Ta＝20～85°C (n＝1～4、m＝1,2)**

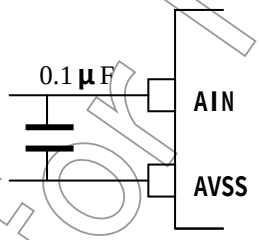
| Parameter | | Symbol | Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Analog reference voltage (+) | | VREFH | | 2.7<br>AVCCm－0.3 | AVCC | 3.3<br>AVCCm＋0.3 | V |
| Analog reference voltage (–) | | VREFL | | AVSS | AVSS | AVSS＋0.2 | |
| Analog input voltage | | VAIN | | VREFL | | VREFH | |
| Analog supply current | During conversion ADMOD1.VREFON＝1 | IREF | AVCCm ＝ VREFH = 3.0V ± 0.3V DVSS = AVSS = VREFL | | 0.35 | 1.0 | mA |
| | Not conversion ADMOD1.VREFON＝0 | | AVCCm ＝ VREFH = 3.0V ± 0.3V DVSS = AVSS = VREFL | | 0.02 | 10 | μA |
| Analog input capacitance | | – | | | 5.0 | | pF |
| Analog input impedance | | – | | | 5.0 | | **kΩ** |
| Integral linearity error | | – | AVCCm ＝ VREFH = 3.0V ± 0.2V DVSS = AVSS = VREFL AIN input impedance R＜13.3kΩ C＜20pF AVCCm capacitor≧10μF VREFH capacitor≧10μF Conversion time≧7.9μs Note* | | ±2 | ± 3 | LSB |
| Differential linearity error | | – | | | ± 1.5 | ± 3 | LSB |
| Off set error | | – | | | ± 2 | ± 3 | LSB |
| Gain error | | – | | | ± 2 | ± 6 | LSB |

Note 1: 1 LSB = (VREFH – VREFL) / 1024 (V)

Note 2: The supply current flowing through the AVCC pin is included in the digital supply current parameter (ICC).

Note 3: Please shift to halt condition in AD converter before changing the standby mode of this product to STOP mode.

Note*:Recommend to connect an external capacitor*



## 4.7 AC Electrical Characteristics

### 4.7.1 Multiplex Bus Mode

(1) DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ± 0.3 V, AVCC3m = 3.3 ± 0.2 V, DVCC33 = 3.0 V ± 0.3 V, Ta = -20 to 85°C (m = 1 to 2)

1. ALE width = 0.5 clock cycle, 1 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A15 valid to ALE low | $t_{AL}$ | 0.5x – 4.3 | | 8 | | ns |
| 3 | A0-A15 hold after ALE low | $t_{LA}$ | 0.5x – 1.8 | | 10.5 | | ns |
| 4 | ALE pulse width high | $t_{LL}$ | 0.5x – 0.3 | | 12 | | ns |
| 5 | ALE low to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{LC}$ | 0.5x – 2.3 | | 10 | | ns |
| 6 | $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated to ALE high | $t_{CL}$ | x – 0.6 | | 24 | | ns |
| 7 | A0-A15 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACL}$ | x – 5.1 | | 19.5 | | ns |
| 8 | A16-A23 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACH}$ | x – 5.1 | | 19.5 | | ns |
| 9 | A16-A23 hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 10 | A0-A15 valid to D0-D15 Data in | $t_{ADL}$ | | x (2 + W) – 35.8 | | 38 | ns |
| 11 | A16-A23 valid to D0-D15 Data in | $t_{ADH}$ | | x (2 + W) – 35.8 | | 38 | ns |
| 12 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 30.7 | | 18.5 | ns |
| 13 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.7 | | 46.5 | | ns |
| 14 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 15 | $\overline{RD}$ negated to next A0-A15 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 16 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 3.2 | | 46 | | ns |
| 17 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 4.2 | | 45 | | ns |
| 18 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x – 0.1 | | 24.5 | | ns |
| 19 | A16-A23 valid to $\overline{WAIT}$ input | $t_{AWH}$ | | x (3 + 0.5) – 21.6 | | 64.5 | ns |
| 20 | A0-A15 valid to $\overline{WAIT}$ input | $t_{AWL}$ | | x (3 + 0.5) – 21.6 | | 64.5 | ns |
| 21 | $\overline{WAIT}$ hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 4.1 | x (1.5 + 3 + N – 2) – 18.7 | 57.4 | 67.4 | ns |

Note: No. 1 to 18 indicate the values obtained with 1 programmed wait state. N0. 19 and 20 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

2. ALE width = 1.5 clock cycles, 1 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A15 valid to ALE low | $t_{AL}$ | 1.5x – 3.9 | | 33 | | ns |
| 3 | A0-A15 hold after ALE low | $t_{LA}$ | 0.5x – 1.8 | | 10.5 | | ns |
| 4 | ALE pulse width high | $t_{LL}$ | 1.5x – 0.4 | | 36.5 | | ns |
| 5 | ALE low to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{LC}$ | 0.5x – 2.4 | | 10 | | ns |
| 6 | $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated to ALE high | $t_{CL}$ | x – 0.6 | | 24 | | ns |
| 7 | A0-A15 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACL}$ | 2x – 5.2 | | 44 | | ns |
| 8 | A16-A23 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACH}$ | 2x – 5.2 | | 44 | | ns |
| 9 | A16-A23 hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 10 | A0-A15 valid to D0-D15 Data in | $t_{ADL}$ | | x (3 + W) – 35.9 | | 62.5 | ns |
| 11 | A16-A23 valid to D0-D15 Data in | $t_{ADH}$ | | x (3 + W) – 35.9 | | 62.5 | ns |
| 12 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 30.7 | | 18.5 | ns |
| 13 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.7 | | 46.5 | | ns |
| 14 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 15 | $\overline{RD}$ negated to next A0-A15 output | $t_{RAE}$ | x | | 24.6 | | ns |
| 16 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 3.2 | | 46 | | ns |
| 17 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 4.2 | | 45 | | ns |
| 18 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x – 0.1 | | 24.5 | | ns |
| 19 | A16-A23 valid to $\overline{WAIT}$ input | $t_{AWH}$ | | x (4 + 0.5) – 21.7 | | 89 | ns |
| 20 | A0-A15 valid to $\overline{WAIT}$ input | $t_{AWL}$ | | x (4 + 0.5) – 21.7 | | 89 | ns |
| 21 | $\overline{WAIT}$ hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 4.1 | x (1.5 + 3 + N – 2) – 18.7 | 57.4 | 67.4 | ns |

Note: No. 1 to 18 indicate the values obtained with 1 programmed wait state. N0. 19 and 20 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

(2) DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ± 0.3 V, AVCC3m = 3.3 ± 0.2 V, DVCC33 = 2.5 V ± 0.2 V, Ta = 20 to 85°C (m = 1 to 2)

1. ALE width = 0.5 clock cycle, 1 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A15 valid to ALE low | $t_{AL}$ | 0.5x – 2.3 | | 10 | | ns |
| 3 | A0-A15 hold after ALE low | $t_{LA}$ | 0.5x – 1.8 | | 10.5 | | ns |
| 4 | ALE pulse width high | $t_{LL}$ | 0.5x – 0.3 | | 12 | | ns |
| 5 | ALE low to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{LC}$ | 0.5x – 2.3 | | 10 | | ns |
| 6 | $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ negated to ALE high | $t_{CL}$ | x – 0.6 | | 24 | | ns |
| 7 | A0-A15 valid to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACL}$ | x – 5.1 | | 19.5 | | ns |
| 8 | A16-A23 valid to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACH}$ | x – 5.1 | | 19.5 | | ns |
| 9 | A16-A23 hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 10 | A0-A15 valid to D0-D15 Data in | $t_{ADL}$ | | x (2 + W) – 36.8 | | 37 | ns |
| 11 | A16-A23 valid to D0-D15 Data in | $t_{ADH}$ | | x (2 + W) – 36.8 | | 37 | ns |
| 12 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 31.7 | | 17.5 | ns |
| 13 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.2 | | 47 | | ns |
| 14 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 15 | $\overline{RD}$ negated to next A0-A15 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 16 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 2.7 | | 46.5 | | ns |
| 17 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 3.8 | | 45.5 | | ns |
| 18 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x –0.1 | | 24.5 | | ns |
| 19 | A16-A23 valid to $\overline{WAIT}$ input | $t_{AWH}$ | | x (3 + 0.5) – 22.6 | | 63.5 | ns |
| 20 | A0-A15 valid to $\overline{WAIT}$ input | $t_{AWL}$ | | x (3 + 0.5) – 22.6 | | 63.5 | ns |
| 21 | $\overline{WAIT}$ hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 5.1 | x (1.5 + 3 + N – 2) – 19.7 | 56.4 | 66.4 | ns |

Note: No. 1 to 18 indicate the values obtained with 1 programmed wait state. N0. 19 and 20 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

2. ALE width = 1.5 clock cycles, 1 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A15 valid to ALE low | $t_{AL}$ | 1.5x – 2.4 | | 34.5 | | ns |
| 3 | A0-A15 hold after ALE low | $t_{LA}$ | 0.5x – 1.8 | | 10.5 | | ns |
| 4 | ALE pulse width high | $t_{LL}$ | 1.5x – 0.4 | | 36.5 | | ns |
| 5 | ALE low to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{LC}$ | 0.5x – 2.4 | | 10 | | ns |
| 6 | $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated to ALE high | $t_{CL}$ | x – 0.6 | | 24 | | ns |
| 7 | A0-A15 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACL}$ | 2x – 5.2 | | 44 | | ns |
| 8 | A16-A23 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACH}$ | 2x – 5.2 | | 44 | | ns |
| 9 | A16-A23 hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 10 | A0-A15 valid to D0-D15 Data in | $t_{ADL}$ | | x (3 + W) – 36.9 | | 61.5 | ns |
| 11 | A16-A23 valid to D0-D15 Data in | $t_{ADH}$ | | x (3 + W) – 36.9 | | 61.5 | ns |
| 12 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 31.7 | | 17.5 | ns |
| 13 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.2 | | 47 | | ns |
| 14 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 15 | $\overline{RD}$ negated to next A0-A15 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 16 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 2.7 | | 46.5 | | ns |
| 17 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 3.8 | | 45.5 | | ns |
| 18 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x –0.1 | | 24.5 | | ns |
| 19 | A16-A23 valid to $\overline{WAIT}$ input | $t_{AWH}$ | | x (4 + 0.5) – 22.6 | | 88.1 | ns |
| 20 | A0-A15 valid to $\overline{WAIT}$ input | $t_{AWL}$ | | x (4 + 0.5) – 22.6 | | 88.1 | ns |
| 21 | $\overline{WAIT}$ hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 5.1 | x (1.5 + 3 + N – 2) – 19.7 | 56.4 | 66.4 | ns |

Note: No. 1 to 18 indicate the values obtained with 1 programmed wait state. N0. 19 and 20 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

(3) DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ± 0.3 V, AVCC3m = 3.3 ± 0.2 V, DVCC33 = 1.8 V ± 0.15 V, Ta = 20 to 85°C (m = 1 to 2)

1. ALE width = 0.5 clock cycle, 2 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | 33333 | | | ns |
| 2 | A0-A15 valid to ALE low | $t_{AL}$ | 0.5x – 2.3 | | 10 | | ns |
| 3 | A0-A15 hold after ALE low | $t_{LA}$ | 0.5x – 1.8 | | 10.5 | | ns |
| 4 | ALE pulse width high | $t_{LL}$ | 0.5x – 0.3 | | 12 | | ns |
| 5 | ALE low to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{LC}$ | 0.5x – 2.3 | | 10 | | ns |
| 6 | $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated to ALE high | $t_{CL}$ | x – 0.6 | | 24 | | ns |
| 7 | A0-A15 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACL}$ | x – 5.1 | | 19.5 | | ns |
| 8 | A16-A23 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACH}$ | x – 5.1 | | 19.5 | | ns |
| 9 | A16-A23 hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 10 | A0-A15 valid to D0-D15 Data in | $t_{ADL}$ | | x (2 + W) – 42.4 | | 56 | ns |
| 11 | A16-A23 valid to D0-D15 Data in | $t_{ADH}$ | | x (2 + W) – 42.4 | | 56 | ns |
| 12 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 37.3 | | 36.5 | ns |
| 13 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.3 | | 71.5 | | ns |
| 14 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 15 | $\overline{RD}$ negated to next A0-A15 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 16 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 2.8 | | 71 | | ns |
| 17 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 3.8 | | 70 | | ns |
| 18 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x – 0.1 | | 24.5 | | ns |
| 19 | A16-A23 valid to $\overline{WAIT}$ input | $t_{AWH}$ | | x (3 + 0.5) – 28.1 | | 58 | ns |
| 20 | A0-A15 valid to $\overline{WAIT}$ input | $t_{AWL}$ | | x (3 + 0.5) – 28.1 | | 58 | ns |
| 21 | $\overline{WAIT}$ hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 6.1 | x (1.5 + 3 + N – 2) – 24.7 | 55.4 | 61.4 | ns |

Note: No. 1 to 18 indicate the values obtained with 1 programmed wait state. N0. 19 and 20 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

2. ALE width = 1.5 clock cycles, 2 programmed wait states

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A15 valid to ALE low | $t_{AL}$ | 1.5x – 2.4 | | 34.5 | | ns |
| 3 | A0-A15 hold after ALE low | $t_{LA}$ | 0.5x – 1.8 | | 10.5 | | ns |
| 4 | ALE pulse width high | $t_{LL}$ | 1.5x – 0.4 | | 36.5 | | ns |
| 5 | ALE low to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{LC}$ | 0.5x – 2.3 | | 10 | | ns |
| 6 | $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated to ALE high | $t_{CL}$ | x – 0.6 | | 24 | | ns |
| 7 | A0-A15 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACL}$ | 2x – 5.2 | | 44 | | ns |
| 8 | A16-A23 valid to $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{ACH}$ | 2x – 5.2 | | 44 | | ns |
| 9 | A16-A23 hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 10 | A0-A15 valid to D0-D15 Data in | $t_{ADL}$ | | x (3 + W) – 42.5 | | 80.5 | ns |
| 11 | A16-A23 valid to D0-D15 Data in | $t_{ADH}$ | | x (3 + W) – 42.5 | | 80.5 | ns |
| 12 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 37.3 | | 36.5 | ns |
| 13 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.3 | | 71.5 | | ns |
| 14 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 15 | $\overline{RD}$ negated to next A0-A15 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 16 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 2.8 | | 71 | | ns |
| 17 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 3.8 | | 70 | | ns |
| 18 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x – 0.1 | | 24.5 | | ns |
| 19 | A16-A23 valid to $\overline{WAIT}$ input | $t_{AWH}$ | | x (4 + 0.5) – 28.1 | | 82.6 | ns |
| 20 | A0-A15 valid to $\overline{WAIT}$ input | $t_{AWL}$ | | x (4 + 0.5) – 28.1 | | 82.6 | ns |
| 21 | $\overline{WAIT}$ hold after $\overline{RD}$ , $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 6.1 | x (1.5 + 3 + N – 2) – 24.7 | 55.4 | 61.4 | ns |

Note: No. 1 to 18 indicate the values obtained with 1 programmed wait state. N0. 19 and 20 indicate the values obtained with 4 externally generated wait states with N = 1.

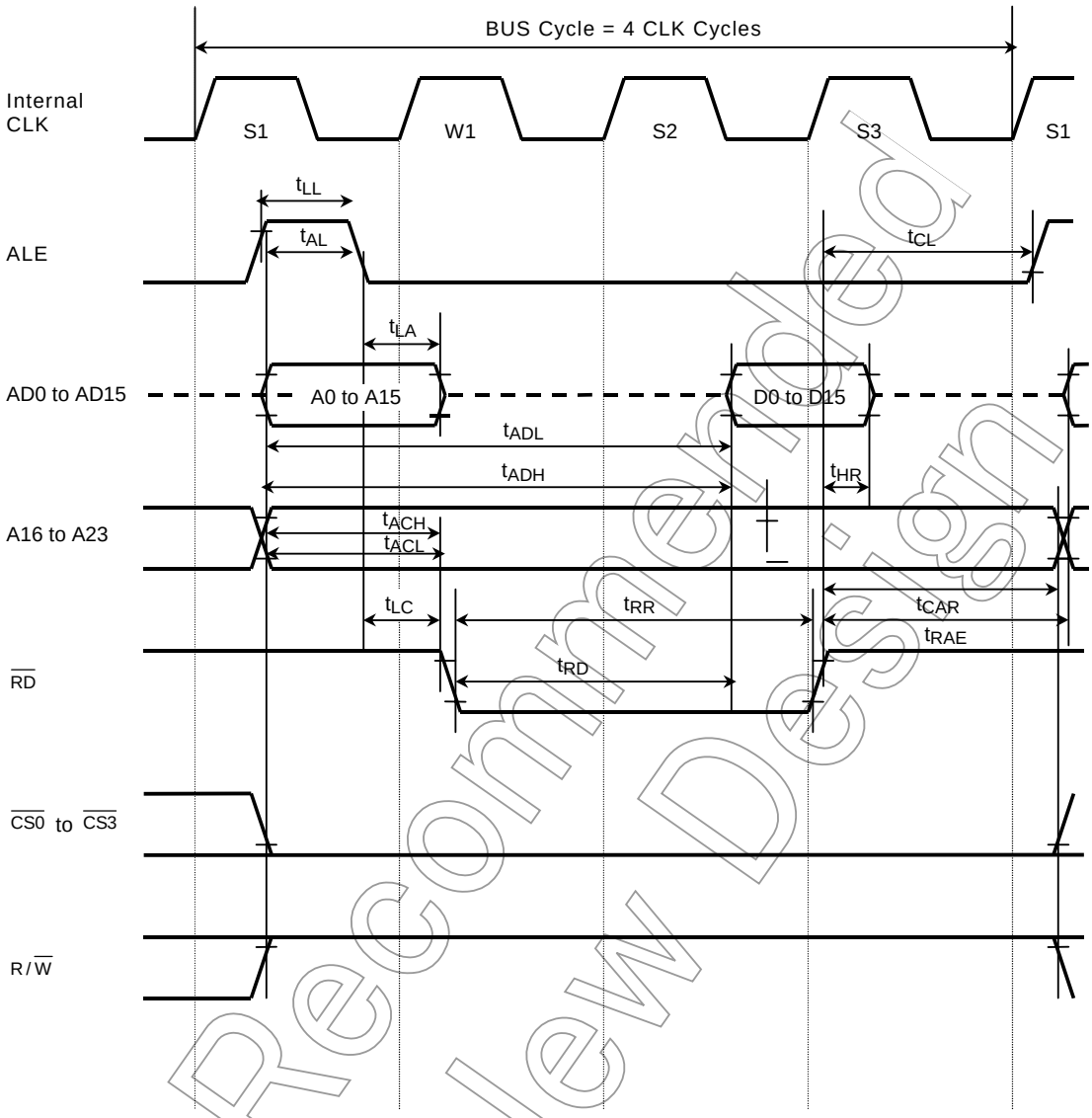
AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

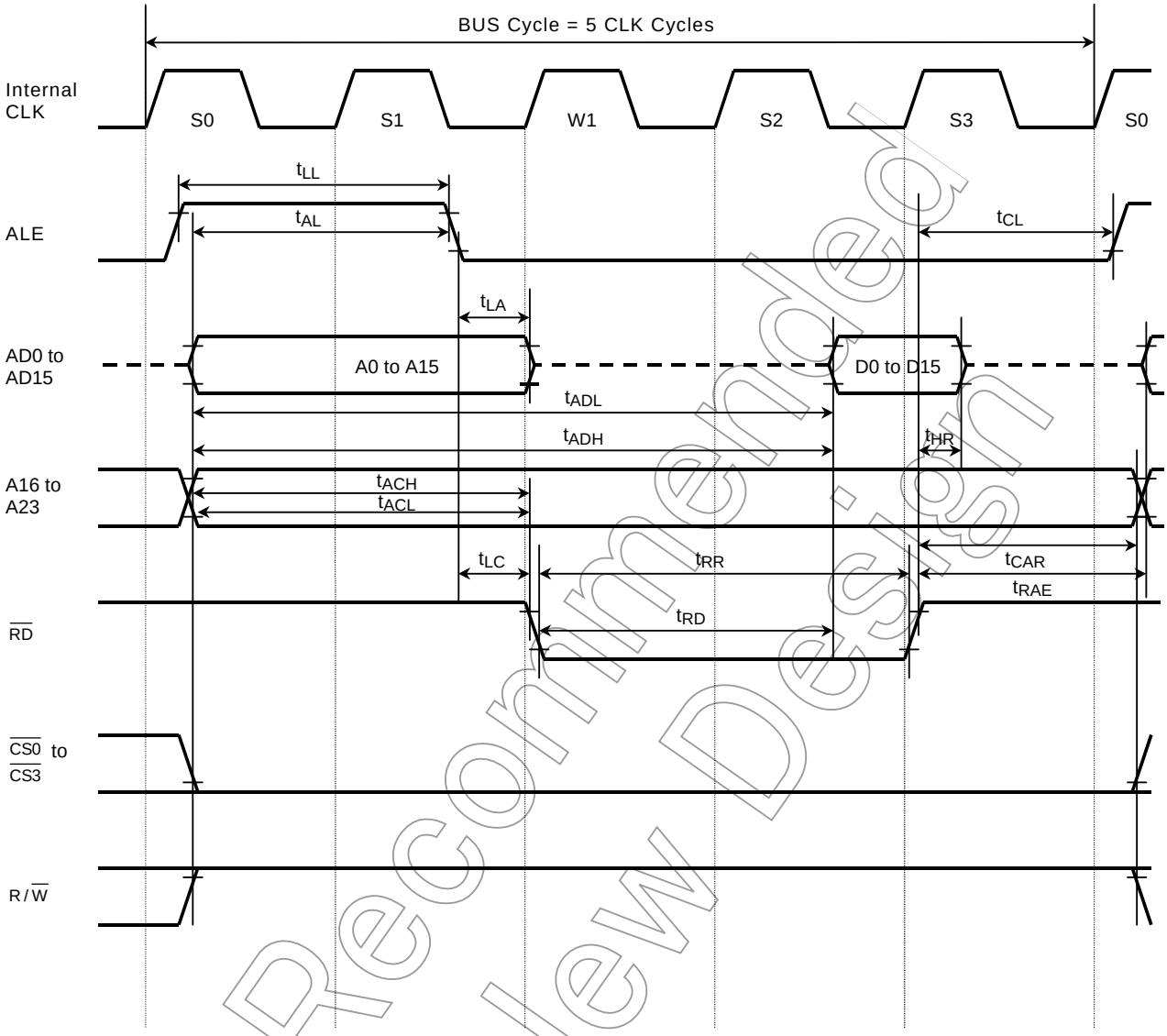
W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

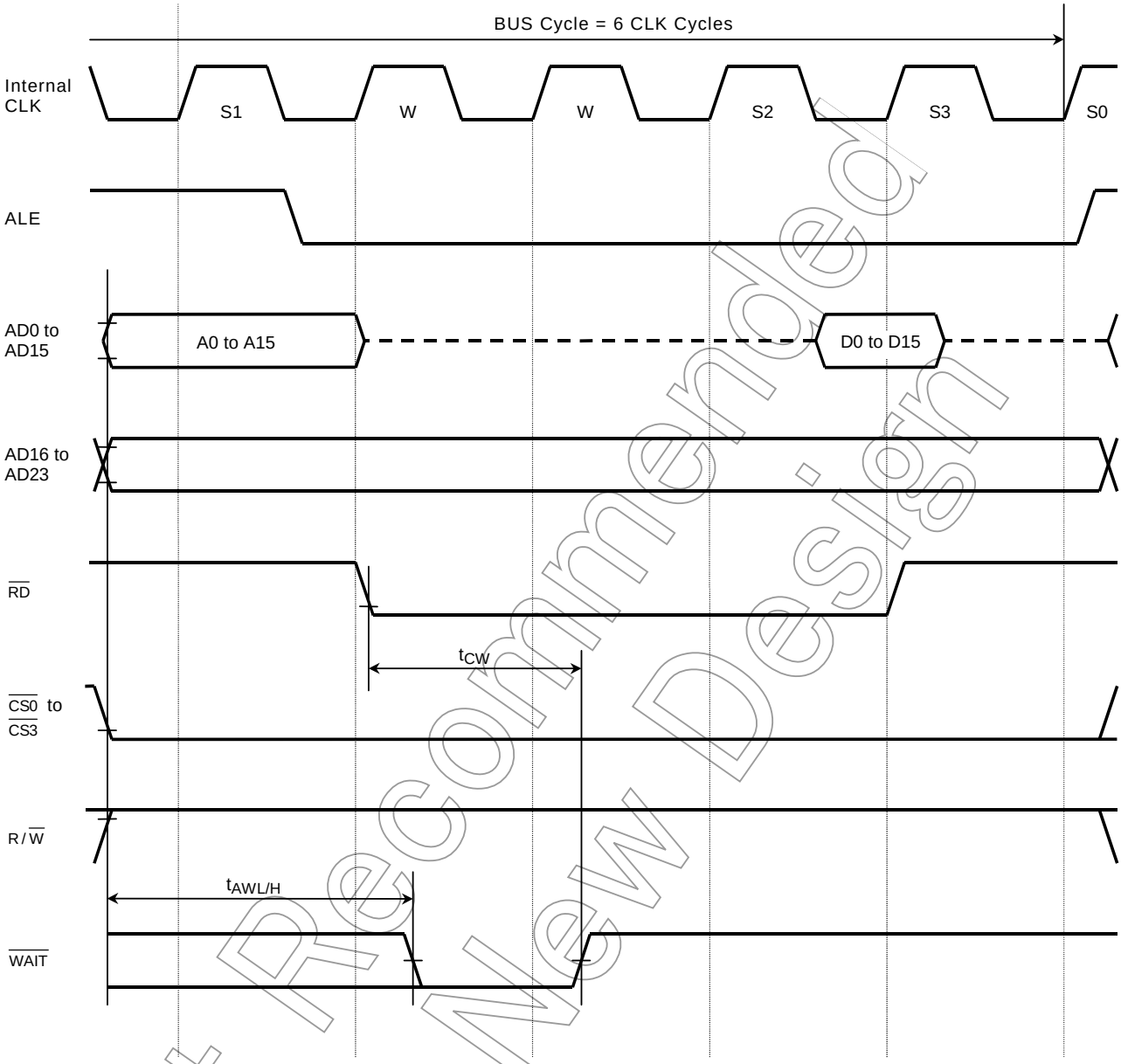
(1) Read cycle timing, ALE width = 0.5 clock cycle, 1 programmed wait state



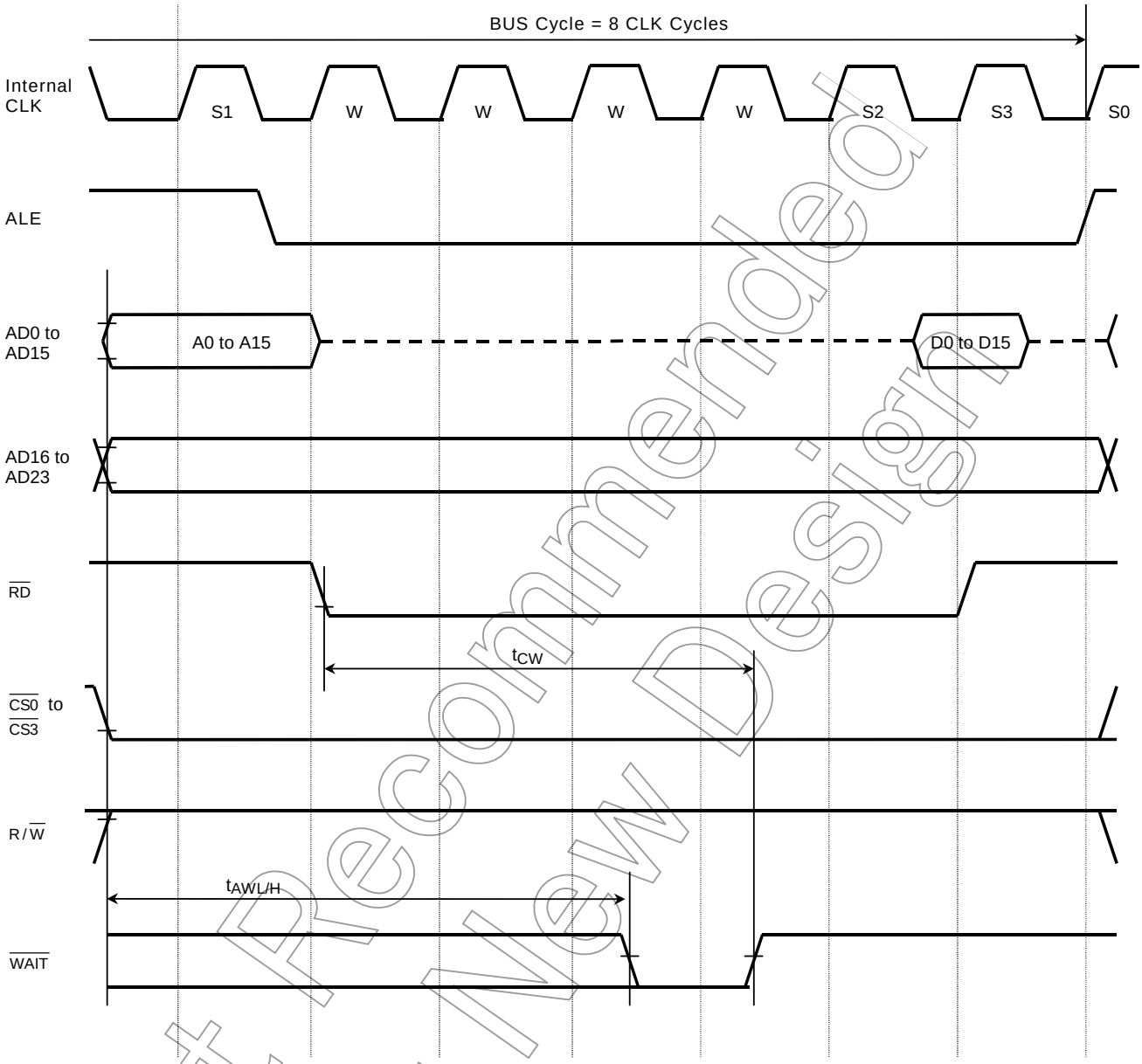
(2) Read cycle timing, ALE width = 1.5 clock cycles, 1 programmed wait state



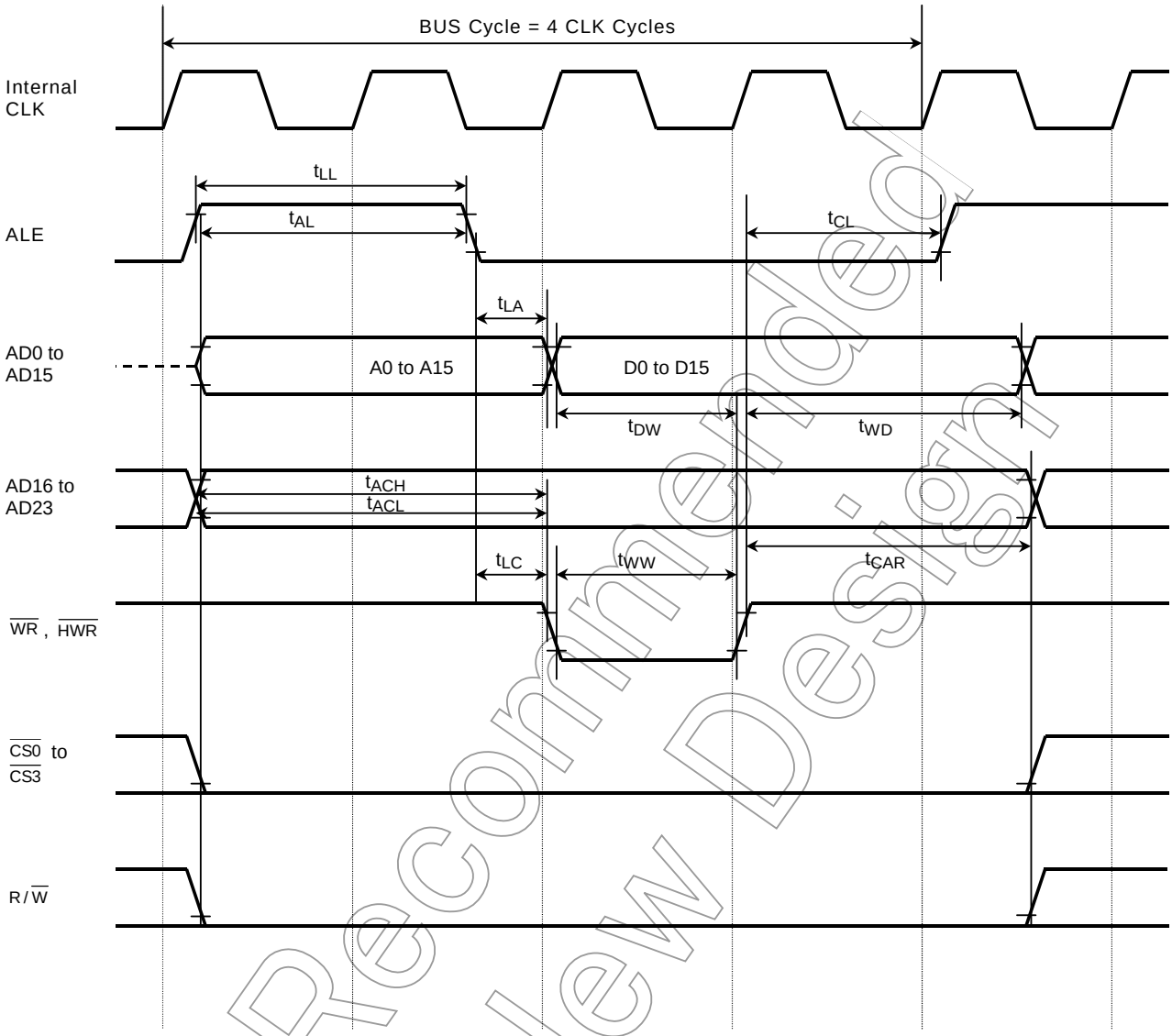
(3) Read cycle timing, ALE width = 1.5 clock cycles, 2 externally generated wait states with N = 1



(4) Read cycle timing, ALE width = 1.5 clock cycles, 4 externally generated wait states with N = 1



(5) Write cycle timing, ALE width = 1.5 clock cycles, zero wait state



### 4.7.2 Separate Bus Mode

(1) DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ± 0.3 V, AVCC3m = 3.3 ± 0.2 V, DVCC33 = 3.0 V ± 0.3 V, Ta = -20 to 85°C (m = 1 to 2)

1. SYSCR3<ALESEL> = 0, 1 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A23 valid to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{AC}$ | x – 5.1 | | 19.5 | | ns |
| 3 | A0-A23 hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 4 | A0-A23 valid to D0-D15 Data in | $t_{AD}$ | | x (2 + W) – 35.8 | | 38 | ns |
| 5 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 30.7 | | 18.5 | ns |
| 6 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.7 | | 46.5 | | ns |
| 7 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 8 | $\overline{RD}$ negated to next A0-A23 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 9 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 3.2 | | 46 | | ns |
| 10 | $\overline{WR}$ or $\overline{HWR}$ asserted to D0-D15 valid | $t_{DO}$ | | — | | 1 | ns |
| 11 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 4.2 | | 45 | | ns |
| 12 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x – 0.1 | | 24.5 | | ns |
| 13 | A0-A23 valid to $\overline{WAIT}$ input | $t_{AW}$ | | x (3 + 0.5) – 21.6 | | 64.5 | ns |
| 14 | $\overline{WAIT}$ hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 4.1 | x (1.5 + 3 + N – 2) – 18.7 | 57.4 | 67.4 | ns |

Note: No. 1 to 12 indicate the values obtained with 1 programmed wait state. N0. 13 and 14 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

2. SYSCR3<ALESEL> = 1, 1 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A23 valid to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{AC}$ | 2x – 5.2 | | 44 | | ns |
| 3 | A0-A23 hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 4 | A0-A23 valid to D0-D15 Data in | $t_{AD}$ | | x (3 + W) – 35.9 | | 62.5 | ns |
| 5 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 30.7 | | 18.5 | ns |
| 6 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.7 | | 46.5 | | ns |
| 7 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 8 | $\overline{RD}$ negated to next A0-A23 output | $t_{RAE}$ | x | | 24.6 | | ns |
| 9 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 3.2 | | 46 | | ns |
| 10 | $\overline{WR}$ or $\overline{HWR}$ asserted to D0-D15 valid | $t_{DO}$ | | — | | 1 | ns |
| 11 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 4.2 | | 45 | | ns |
| 12 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x – 0.1 | | 24.5 | | ns |
| 13 | A0-A23 valid to $\overline{WAIT}$ input | $t_{AW}$ | | x (4 + 0.5) – 21.7 | | 89 | ns |
| 14 | $\overline{WAIT}$ hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 4.1 | x (1.5 + 3 + N – 2) – 18.7 | 57.4 | 67.4 | ns |

Note: No. 1 to 12 indicate the values obtained with 1 programmed wait state. N0. 13 and 14 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

(2) DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ± 0.3 V, AVCC3m = 3.3 ± 0.2 V, DVCC33 = 2.5 V ± 0.2 V, Ta = -20 to 85°C (m = 1 to 2)

1. SYSCR3<ALESEL> = 0, 1 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A23 valid to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{AC}$ | x – 5.1 | | 19.5 | | ns |
| 3 | A0-A23 hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 4 | A16-A23 valid to D0-D15 Data in | $t_{AD}$ | | x (2 + W) – 36.8 | | 37 | ns |
| 5 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 31.7 | | 17.5 | ns |
| 6 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.2 | | 47 | | ns |
| 7 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 8 | $\overline{RD}$ negated to next A0-A23 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 9 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 2.7 | | 46.5 | | ns |
| 10 | $\overline{WR}$ or $\overline{HWR}$ asserted to D0-D15 valid | $t_{DO}$ | | — | | 1.5 | ns |
| 11 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 3.8 | | 45.5 | | ns |
| 12 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x –0.1 | | 24.5 | | ns |
| 13 | A0-A23 valid to $\overline{WAIT}$ input | $t_{AW}$ | | x (3 + 0.5) – 22.6 | | 63.5 | ns |
| 14 | $\overline{WAIT}$ hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 5.1 | x (1.5 + 3 + N – 2) – 19.7 | 56.4 | 66.4 | ns |

Note: No. 1 to 12 indicate the values obtained with 1 programmed wait state. N0. 13 and 14 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

2. SYSCR3<ALESEL> = 1, 1 programmed wait state

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A23 valid to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{AC}$ | 2x – 5.2 | | 44 | | ns |
| 3 | A0-A23 hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 4 | A16-A23 valid to D0-D15 Data in | $t_{AD}$ | | x (3 + W) – 36.9 | | 61.5 | ns |
| 5 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 31.7 | | 17.5 | ns |
| 6 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.2 | | 47 | | ns |
| 7 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 8 | $\overline{RD}$ negated to next A0-A23 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 9 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 2.7 | | 46.5 | | ns |
| 10 | $\overline{WR}$ or $\overline{HWR}$ asserted to D0-D15 valid | $t_{DO}$ | | — | | 1.5 | ns |
| 11 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 3.8 | | 45.5 | | ns |
| 12 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x –0.1 | | 24.5 | | ns |
| 13 | A0-A23 valid to $\overline{WAIT}$ input | $t_{AW}$ | | x (4 + 0.5) – 22.6 | | 88.1 | ns |
| 14 | $\overline{WAIT}$ hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 5.1 | x (1.5 + 3 + N – 2) – 19.7 | 56.4 | 66.4 | ns |

Note: No. 1 to 12 indicate the values obtained with 1 programmed wait state. N0. 13 and 14 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

(3) DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ± 0.3 V, AVCC3m = 3.3 ± 0.2 V, DVCC33 = 1.8 V ± 0.15 V, Ta = -20 to 85°C (m = 1 to 2)

1. SYSCR3<ALESEL> = 0, 2 programmed wait states

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A23 valid to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{AC}$ | x – 5.1 | | 19.5 | | ns |
| 3 | A0-A23 hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 4 | A0-A23 valid to D0-D15 Data in | $t_{AD}$ | | x (2 + W) – 42.4 | | 56 | ns |
| 5 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 37.3 | | 36.5 | ns |
| 6 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.3 | | 71.5 | | ns |
| 7 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 8 | $\overline{RD}$ negated to next A0-A23 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 9 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 2.8 | | 71 | | ns |
| 10 | $\overline{WR}$ or $\overline{HWR}$ asserted to D0-D15 valid | $t_{DO}$ | | — | | 2 | ns |
| 11 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 3.8 | | 70 | | ns |
| 12 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x – 0.1 | | 24.5 | | ns |
| 13 | A0-A23 valid to $\overline{WAIT}$ input | $t_{AWH}$ | | x (3 + 0.5) – 28.1 | | 58 | ns |
| 14 | $\overline{WAIT}$ hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 6.1 | x (1.5 + 3 + N – 2) – 24.7 | 55.4 | 61.4 | ns |

Note: No. 1 to 12 indicate the values obtained with 1 programmed wait state. N0. 13 and 14 indicate the values obtained with 4 externally generated wait states with N = 1.

AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

2. SYSCR3<ALESEL> = 1, 2 programmed states

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | System clock period (x) | $t_{SYS}$ | 24.6 | | | | ns |
| 2 | A0-A23 valid to $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{AC}$ | 2x – 5.2 | | 44 | | ns |
| 3 | A0-A23 hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ negated | $t_{CAR}$ | x – 1.6 | | 23 | | ns |
| 4 | A16-A23 valid to D0-D15 Data in | $t_{AD}$ | | x (3 + W) – 42.5 | | 80.5 | ns |
| 5 | $\overline{RD}$ asserted to D0-D15 data in | $t_{RD}$ | | x (1 + W) – 37.3 | | 36.5 | ns |
| 6 | $\overline{RD}$ width low | $t_{RR}$ | x (1 + W) – 2.3 | | 71.5 | | ns |
| 7 | D0-D15 hold after $\overline{RD}$ negated | $t_{HR}$ | 0 | | 0 | | ns |
| 8 | $\overline{RD}$ negated to next A0-A23 output | $t_{RAE}$ | x – 0.1 | | 24.5 | | ns |
| 9 | $\overline{WR}$ or $\overline{HWR}$ width low | $t_{WW}$ | x (1 + W) – 2.8 | | 71 | | ns |
| 10 | $\overline{WR}$ or $\overline{HWR}$ asserted to D0-D15 valid | $t_{DO}$ | | — | | 2 | ns |
| 11 | D0-D15 valid to $\overline{WR}$ or $\overline{HWR}$ negated | $t_{DW}$ | x (1 + W) – 3.8 | | 70 | | ns |
| 12 | D0-D15 hold after $\overline{WR}$ or $\overline{HWR}$ negated | $t_{WD}$ | x – 0.1 | | 24.5 | | ns |
| 13 | A0-A23 valid to $\overline{WAIT}$ input | $t_{AWH}$ | | x (4 + 0.5) – 28.1 | | 82.6 | ns |
| 14 | $\overline{WAIT}$ hold after $\overline{RD}$, $\overline{WR}$ or $\overline{HWR}$ asserted | $t_{CW}$ | x (0.5 + 3 + N – 2) – 6.1 | x (1.5 + 3 + N – 2) – 24.7 | 55.4 | 61.4 | ns |

Note: No. 1 to 12 indicate the values obtained with 1 programmed wait state. N0. 13 and 14 indicate the values obtained with 4 externally generated wait states with N = 1.

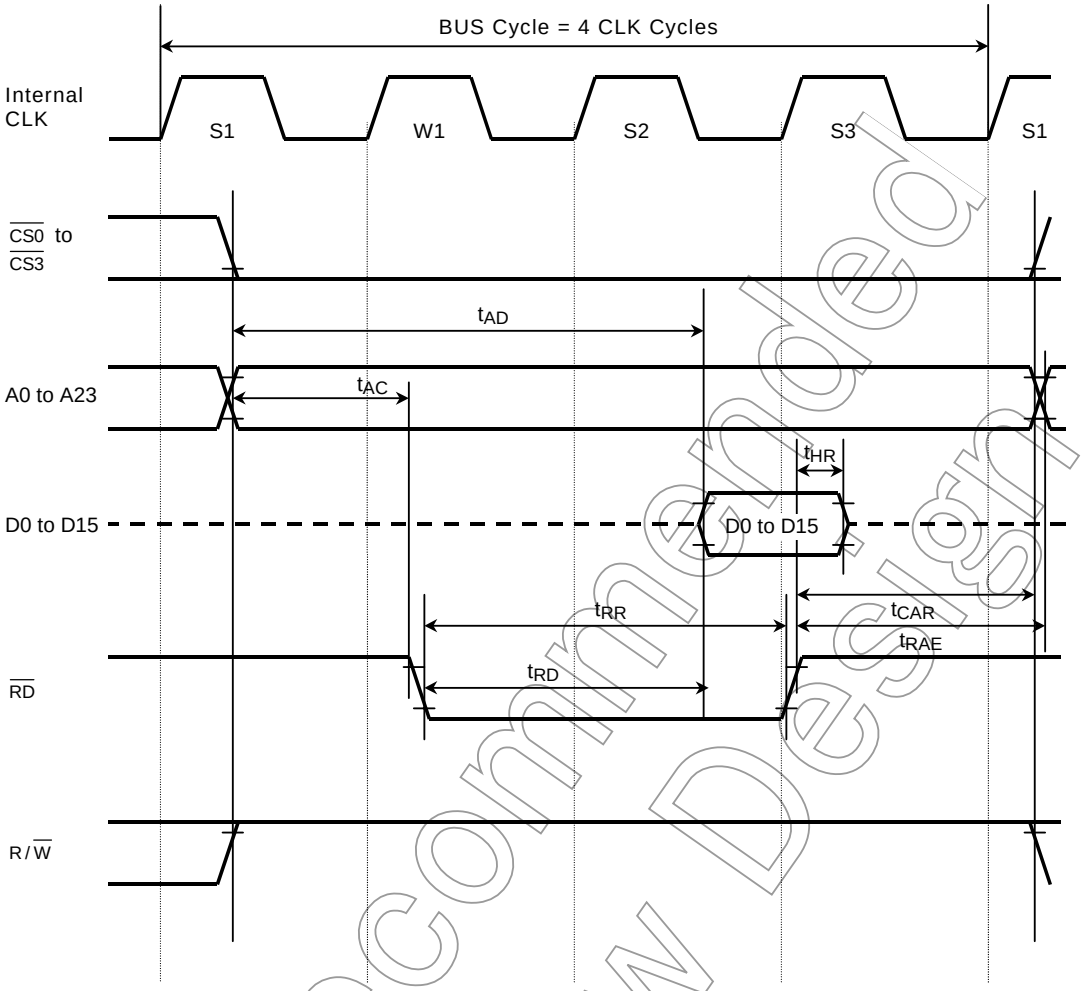
AC measurement conditions:

- Output levels: High = 0.8DVCC33 V/Low 0.2DVCC33 V, CL = 30 pF
- Input levels: High = 0.7DVCC33 V/Low 0.3DVCC33 V

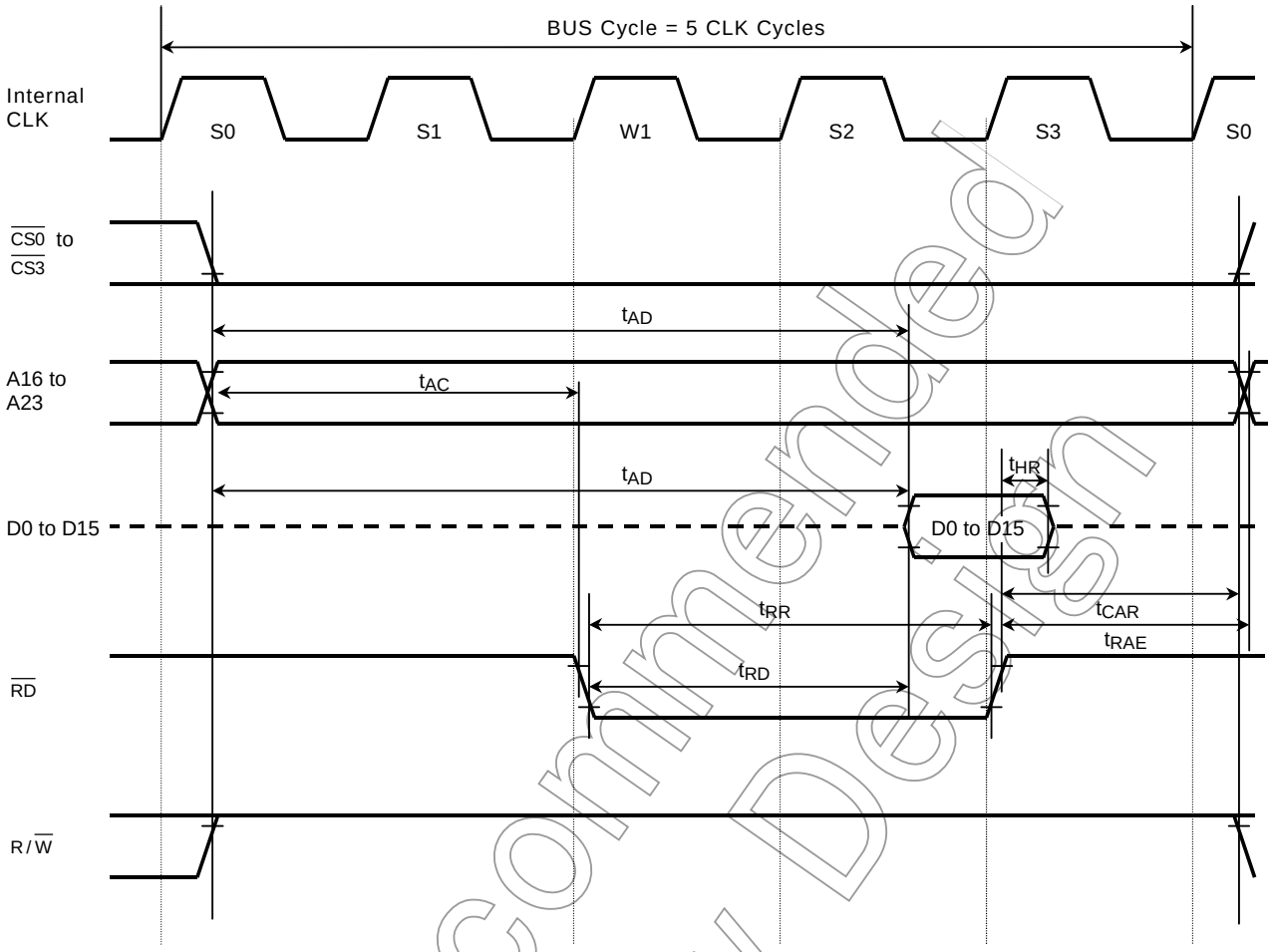
W: Number of wait state cycles inserted (0 to 7 for programmed wait insertion)

N: Value of N for (3 + N) wait insertion

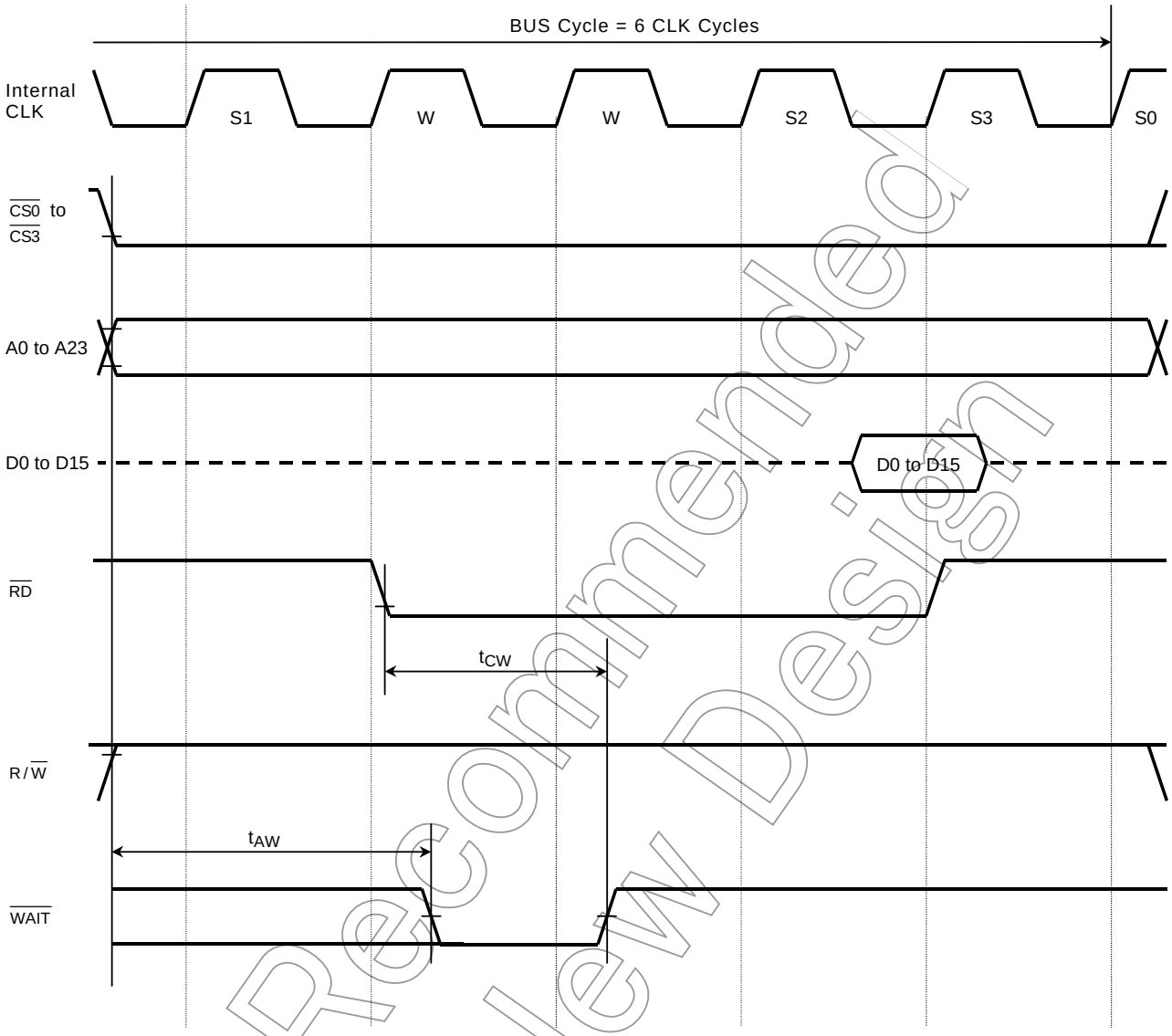
(1) Read cycle timing (SYSCR3<ALESEL> = 0, 1 programmed wait state)



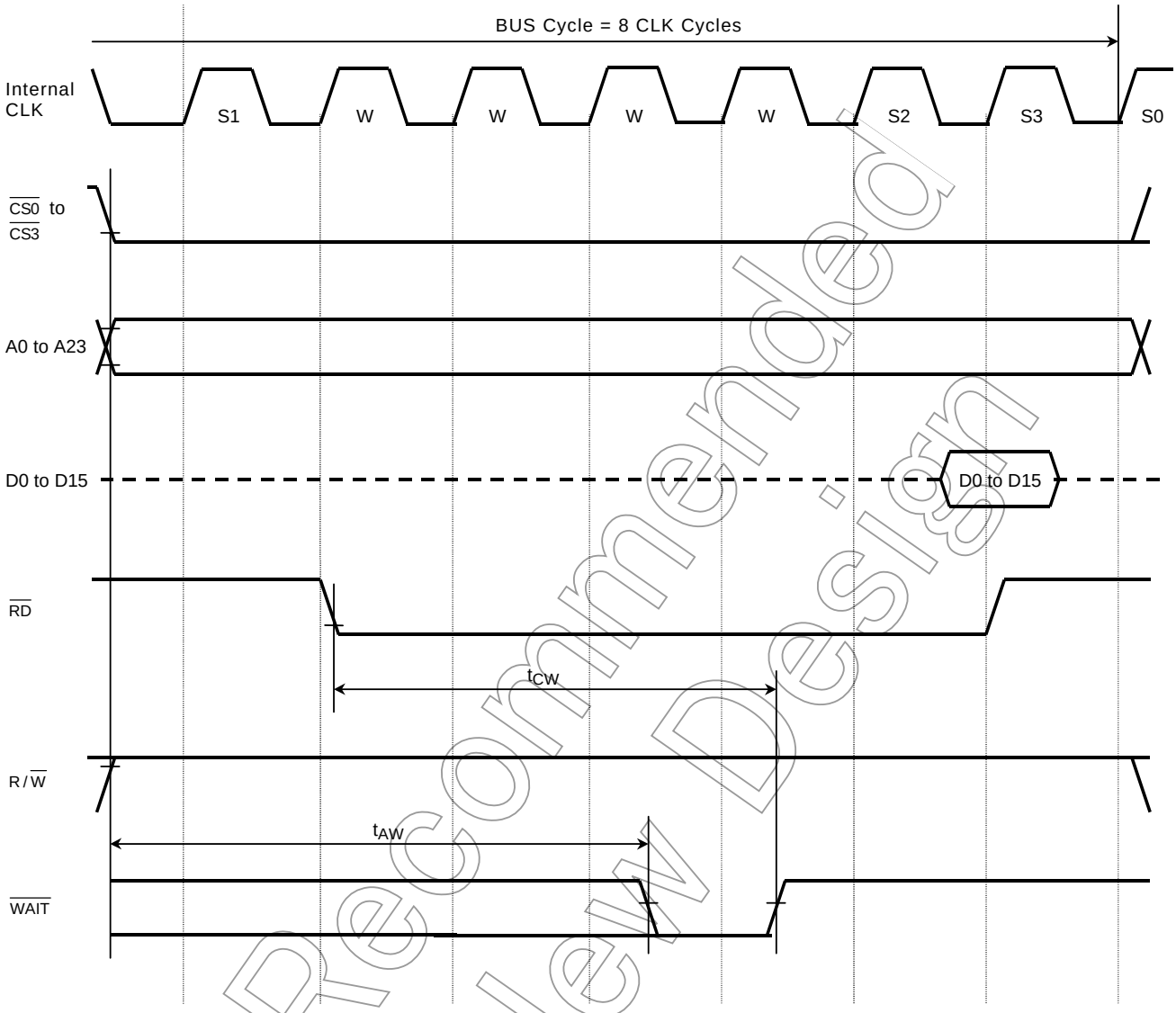
(2) Read cycle timing (SYSCR3<ALESEL> = 1, 1 programmed wait state)



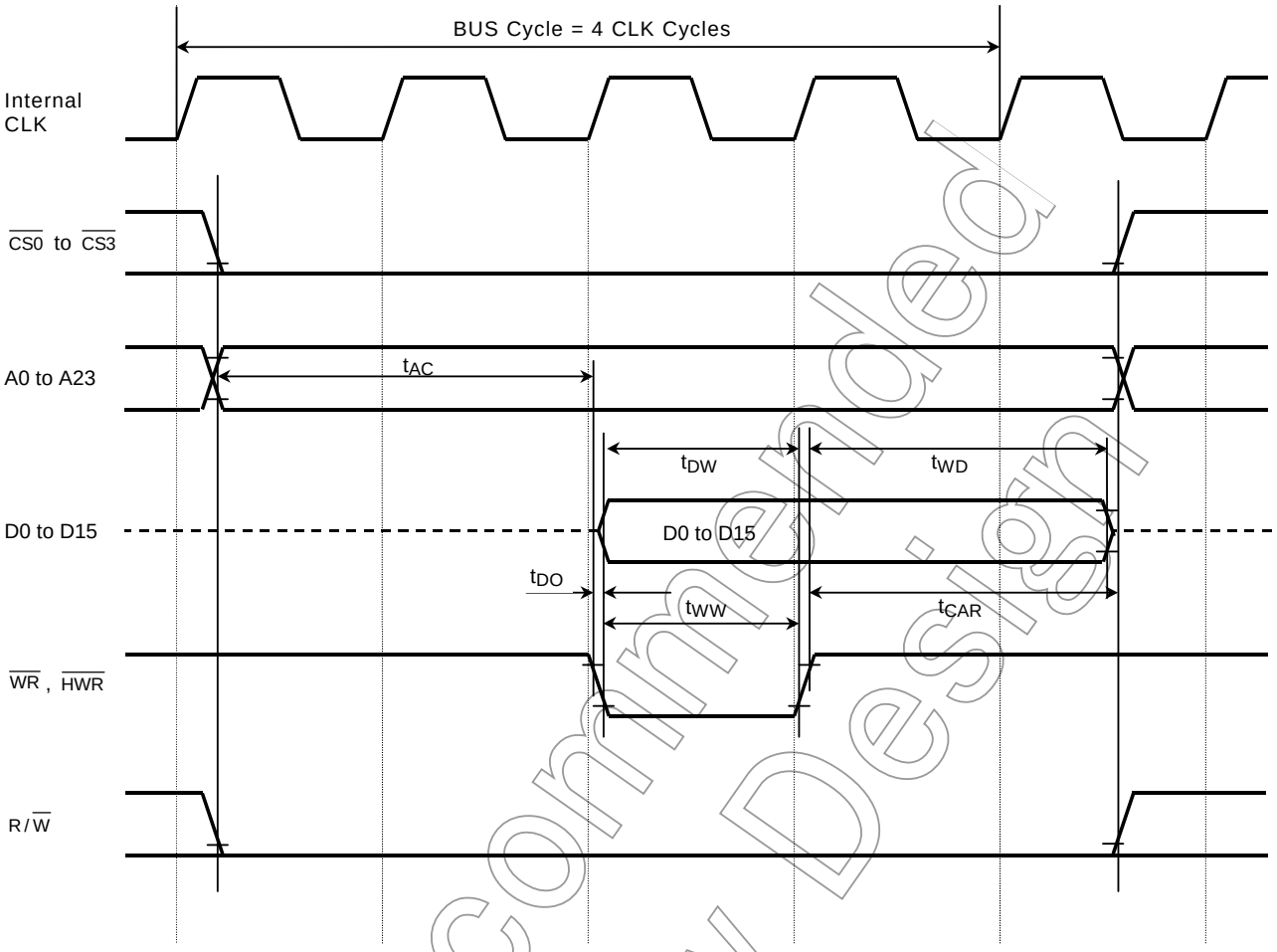
(3) Read cycle timing SYSCR3<ALESEL> = 1, 2 externally generated wait states with N = 1)



(4) Read cycle timing (SYSCR3<ALESEL> = 1, 4 externally generated wait states with N = 1)



(5) Write cycle timing (SYSCR3<ALESEL> = 1, zero wait sate)

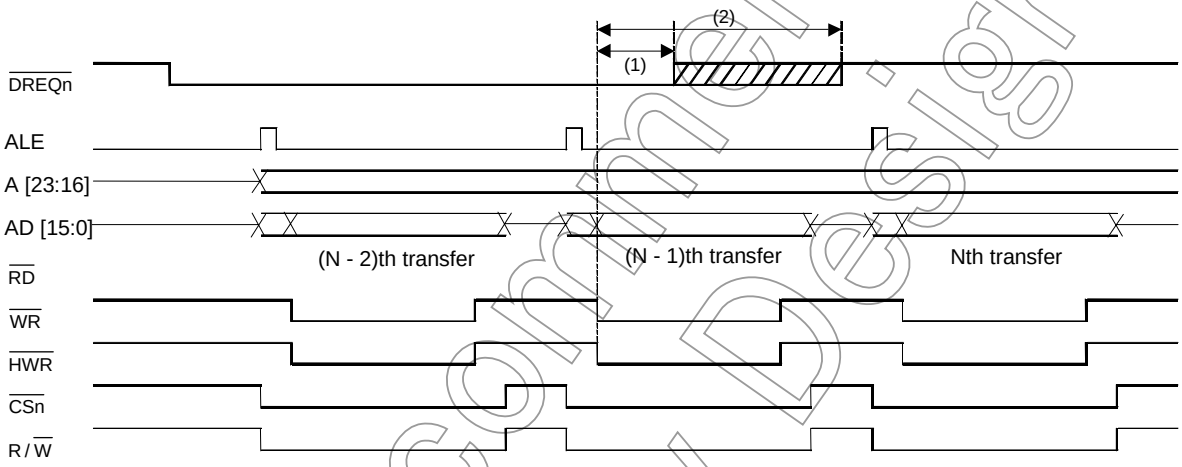


## 4.8 Transfer with DMA Request

The following shows an example of a transfer between the on-chip RAM and an external device in multiplex bus mode.

- 16-bit data bus width, non-recovery time
- Level data transfer mode
- Transfer size of 16 bits, device port size (DPS) of 16 bits
- Source/destination: on-chip RAM/external device

The following shows transfer operation timing of the on-chip RAM to an external bus during write operation (memory-to-memory transfer).




(1) Indicates the condition under which Nth transfer is performed successfully.

(2) Indicates the condition under which (N + 1)th transfer is not performed.

(1) DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ±0.3 V, AVCC3m = 3.3 ± 0.2 V, DVCC33 = 2.3 V to 3.3 V, Ta = -20 to 85°C (m = 1 to 2)

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | (1) Min | (2) Max | Min | Max | |
| 2 | $\overline{RD}$ asserted to $\overline{DREQn}$ negated (external device to on-chip RAM transfer) | tDREQ_r | Wx – 4.2 | (2W + ALE + 6) x – 51 | 45 | 195 | ns |
| 3 | $\overline{WR}$ / $\overline{HWR}$ rising to $\overline{DREQn}$ negated (on-chip RAM to external device transfer) | tDREQ_w | 0 | (2W + ALE + 4) x – 51.8 | 0 | 145 | ns |

(2) DVCC2m = FVCC2 = CVCC2 = 2.5 V ± 0.2 V, FVCC3 = 3.3 V ± 0.3 V, AVCC3m = 3.3 ± 0.2 V, DVCC33 = 1.8 V ± 0.15 V, Ta = 20 to 85°C (m = 1 to 2)

| No. | Parameter | Symbol | Equation | | 40.5 MHz (fsys) (Note) | | Unit |
|---|---|---|---|---|---|---|---|
| | | | (1) Min | (2) Max | Min | Max | |
| 2 | $\overline{RD}$ asserted to $\overline{DREQn}$ negated (external device to on-chip RAM transfer) | tDREQ_r | Wx – 6.2 | (2W + ALE + 6) x – 56 | 43 | 190 | ns |
| 3 | $\overline{WR}$ / $\overline{HWR}$ rising to $\overline{DREQn}$ negated (on-chip RAM to external device transfer) | tDREQ_w | 0 | (2W + ALE + 4) x – 56.8 | 0 | 140 | ns |

W: Number of wait-state cycles inserted. In the case of (1 + N) externally generated wait states with N = 1, W becomes 2.

ALE: Apply ALE = 0 for ALE 0.5 clock, ALE = 1 for ALE 1.5 clock. The values in the above table are obtained with W = 2, ALE = 0.

## 4.9 Serial Channel Timing

(1) I/O Interface Mode (DVCC3n = 3.0 V ± 0.3V)

In the table below, the letter x represents the fsys cycle period, which varies depending on the programming of the clock gear function.

1. SCLK input mode (SIO0 to SIO6)

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| SCLK period | $t_{SCY}$ | 12x | | 296 | | ns |
| TxD data to SCLK rise or fall* | $t_{OSS}$ | 2x – 45 | | 4 | | ns |
| TxD data hold after SCLK rise or fall* | $t_{OHS}$ | 8x – 15 | | 182 | | ns |
| RxD data valid to SCLK rise or fall* | $t_{SRD}$ | 30 | | 30 | | ns |
| RxD data hold after SCLK rise or fall* | $t_{HSR}$ | 2x – 30 | | 19 | | ns |

* SCLK rise or fall: Measured relative to the programmed active edge of SCLK.

2. SCLK output mode (SIO0 to SIO6)

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| SCLK period (programmable) | $t_{SCY}$ | 8x | | 197 | | ns |
| TxD data to SCLK rise | $t_{OSS}$ | 4x – 10 | | 88 | | ns |
| TxD data hold after SCLK rise | $t_{OHS}$ | 4x – 10 | | 88 | | ns |
| RxD data valid to SCLK rise | $t_{SRD}$ | 45 | | 45 | | ns |
| RxD data hold after SCLK rise | $t_{HSR}$ | 0 | | 0 | | ns |

(2) I/O Interface Mode (DVCC3n = 2.5 V ± 0.2 V)

In the table below, the letter x represents the fsys cycle period, which varies depending on the programming of the clock gear function.

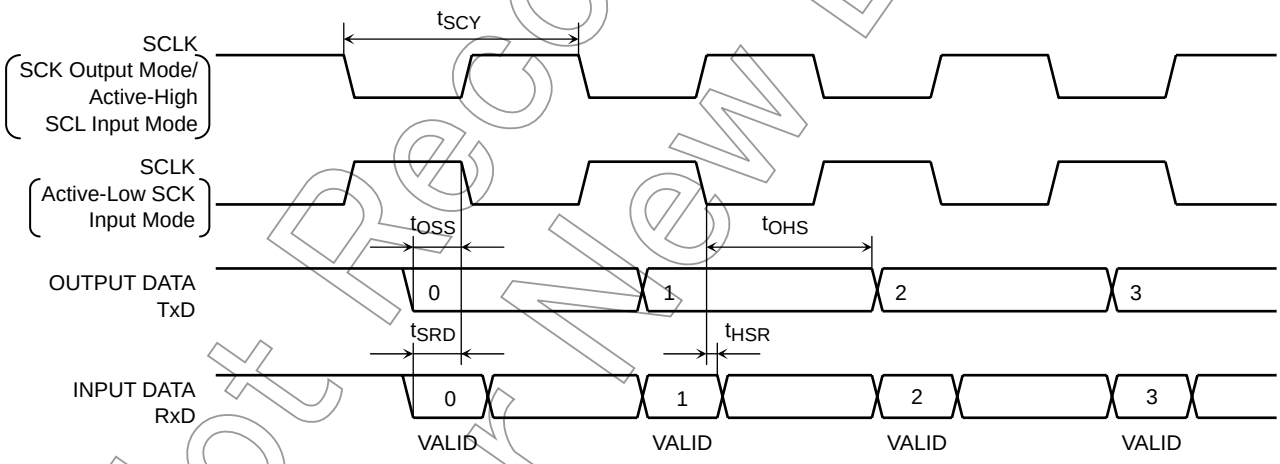
1. SCLK input mode (SIO0 to SIO6)

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| SCLK period | $t_{SCY}$ | 16x | | 395 | | ns |
| TxD data to SCLK rise or fall* | $t_{OSS}$ | 4x – 60 | | 38 | | ns |
| TxD data hold after SCLK rise or fall* | $t_{OHS}$ | 10x – 15 | | 232 | | ns |
| RxD data valid to SCLK rise or fall* | $t_{SRD}$ | 30 | | 30 | | ns |
| RxD data hold after SCLK rise or fall* | $t_{HSR}$ | 2x + 10 | | 59 | | ns |

* SCLK rise or fall: Measured relative to the programmed active edge of SCLK.

2. SCLK output mode (SIO0 to SIO6)

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| SCLK period (programmable) | $t_{SCY}$ | 8x | | 197 | | ns |
| TxD data to SCLK rise | $t_{OSS}$ | 4x – 10 | | 88 | | ns |
| TxD data hold after SCLK rise | $t_{OHS}$ | 4x – 10 | | 88 | | ns |
| RxD data valid to SCLK rise | $t_{SRD}$ | 60 | | 60 | | ns |
| RxD data hold after SCLK rise | $t_{HSR}$ | 0 | | 0 | | ns |



## 4.10 SBI Timing

(1) I2C Mode

In the table below, the letters x and T represent the fsys and ϕT0 cycle periods, respectively. The letter n denotes the value of n programmed into the SCK[2:0] (SCL output frequency select) field in the SBI0CR1.

| Parameter | Symbol | Equation | | Standard Mode $f_{SYS}$ = 8 MHz n = 4 | | Fast Mode $f_{SYS}$ = 32 MHz n = 4 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| SCL clock frequency | $t_{SCL}$ | 0 | | 0 | 100 | 0 | 400 | kHz |
| Hold time for START condition | $t_{HD:STA}$ | | | 4.0 | | 0.6 | | μs |
| SCL clock low width (Input) (Note 1) | $t_{LOW}$ | | | 4.7 | | 1.3 | | μs |
| SCL clock high width (Output) (Note 2) | $t_{HIGH}$ | | | 4.0 | | 0.6 | | μs |
| Setup time for a repeated START condition | $t_{SU:STA}$ | (Note 5) | | 4.7 | | 0.6 | | μs |
| Data hold time (Input) (Note 3, 4) | $t_{HD:DAT}$ | | | 0.0 | | 0.0 | | μs |
| Data setup time | $t_{SU:DAT}$ | | | 250 | | 100 | | ns |
| Setup time for STOP condition | $t_{SU:STO}$ | | | 4.0 | | 0.6 | | μs |
| Bus free time between STOP and START conditions | $t_{BUF}$ | (Note 5) | | 4.7 | | 1.3 | | μs |

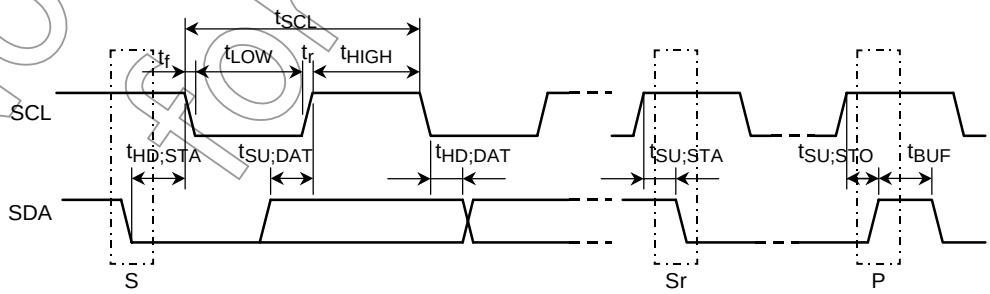
Note 1: SCL clock low width (output) is calculated with (2 (n − 1) + 4) T.
Standard mode: 6 μsec@Typ (fsys = 8 MHz, n = 4)
Fast mode: 1.5 μsec@Typ (fsys = 32 MHz, n = 4)

Note 2: SCL clock high width (output) is calculated with (2 (n − 1)) T.
Standard mode: 4 μsec@Typ (fsys = 8MHz, n = 4)
Fast mode: 1 μsec@Typ (fsys = 32 MHz, n = 4)

Note 3: The output data hold time is equal to 12x.

Note 4: The Philips $I^2C$-bus specification states that a device must internally provide a hold time of at least 300 ns for the SDA signal to bridge the undefined region of the fall edge of SCL. However, TMP1962F10AXBG SBI does not satisfy this requirement. Also, the output buffer for SCL does not incorporate slope control of the falling edges; therefore, the equipment manufacturer should design so that the input data hold time shown in the table is satisfied, including tr/tf of the SCL and SDA lines.

Note 5: Software-dependent




S: START condition
Sr: Repeated START condition
P: STOP condition

Note 6: To operate the SBI in $I^2C$ Fast mode, the fysy frequency must be no less than 20 MHz. To operate the SBI in $I^2C$ Standard mode, the fsys frequency must be no less than 4 MHz.

(2) Clock-Synchronous 8-Bit SIO Mode

In the table below, the letters x and T represent the fsys and ϕT0 cycle periods, respectively. The letter n denotes the value of n programmed into the SCK[2:0] (SCL output frequency select) field in the SBI0CR1.

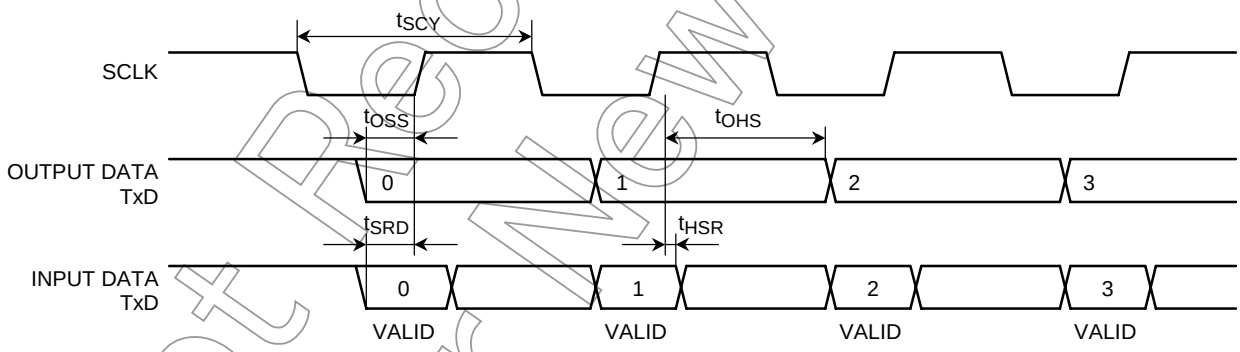
The electrical specifications below are for an SCK signal with a 50% duty cycle.

1. SCK Input Mode

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| SCK period | $t_{SCY}$ | 16x | | 395 | | ns |
| SO data to SCK rise | $t_{OSS}$ | $(t_{SCY}/2) - (6x + 30)$ | | 19 | | ns |
| SO data hold after SCK rise | $t_{OHS}$ | $(t_{SCY}/2) + 4x$ | | 296 | | ns |
| SI data valid to SCK rise | $t_{SRD}$ | 0 | | 0 | | ns |
| SI data hold after SCK rise | $t_{HSR}$ | 4x + 10 | | 108 | | ns |

2. SCK Output Mode

| Parameter | Symbol | Equation | | 32 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| SCK period (programmable) | $t_{SCY}$ | $2^n \cdot T$ | | 1000 | | ns |
| SO data to SCK rise | $t_{OSS}$ | $(t_{SCY}/2) - 20$ | | 480 | | ns |
| SO data hold after SCK rise | $t_{OHS}$ | $(t_{SCY}/2) - 20$ | | 480 | | ns |
| SI data valid to SCK rise | $t_{SRD}$ | 2x + 30 | | 92 | | ns |
| SI data hold after SCK rise | $t_{HSR}$ | 0 | | 0 | | ns |



## 4.11 Event Counter

In the table below, the letter x represents the fsys cycle period.

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Clock low pulse width | $t_{VCKL}$ | 2X + 100 | | 149 | | ns |
| Clock high pulse width | $t_{VCKH}$ | 2X + 100 | | 149 | | ns |

## 4.12 Timer Capture

In the table below, the letter x represents the fsys cycle period.

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Low pulse width | $t_{CPL}$ | 2X + 100 | | 149 | | ns |
| High pulse width | $t_{CPH}$ | 2X + 100 | | 149 | | ns |

## 4.13 General Interrupts

In the table below, the letter x represents the fsys cycle period.

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Low pulse width for INT0-INTA | $t_{INTAL}$ | X + 100 | | 125 | | ns |
| High pulse width for INT0-INTA | $t_{INTAH}$ | X + 100 | | 125 | | ns |

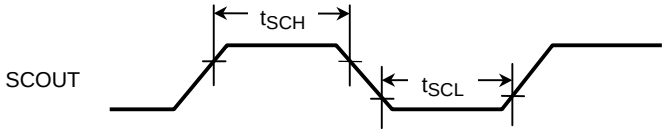
## 4.14 $\overline{NMI}$ and STOP Wake-up Interrupts

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Low pulse width for $\overline{NMI}$ and INT0-INT4 | $t_{INTBL}$ | 100 | | 100 | | ns |
| High pulse width for INT0-INT4 | $t_{INTBH}$ | 100 | | 100 | | ns |

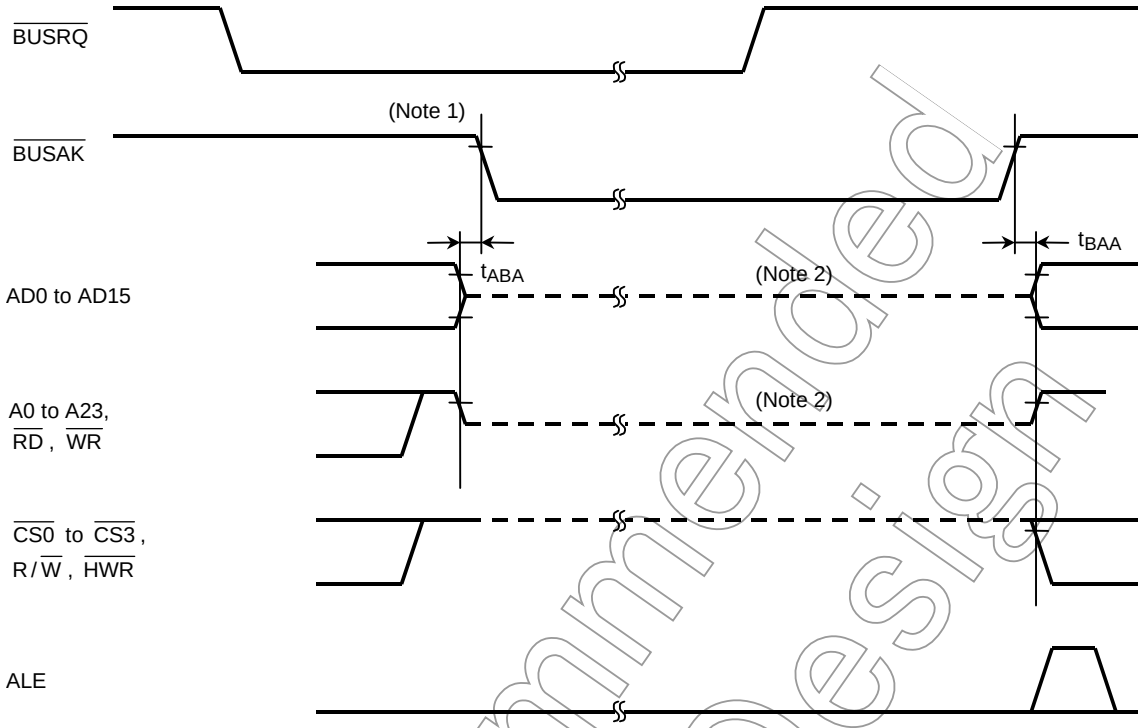
## 4.15 SCOUT Pin

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Clock high pulse width | $t_{SCH}$ | 0.5T – 5 | | 7.4 | | ns |
| Clock low pulse width | $t_{SCL}$ | 0.5T – 5 | | 7.4 | | ns |

Note: In the above table, the letter T represents the cycle period of the SCOUT output clock.



## 4.16 Bus Request and Bus Acknowledge Signals




| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Bus float to $\overline{BUSAK}$ asserted | $t_{ABA}$ | 0 | 80 | 0 | 80 | ns |
| Bus float after $\overline{BUSAK}$ negated | $t_{BAA}$ | 0 | 80 | 0 | 80 | ns |

Note 1: If the current bus cycle has not terminated due to wait-state insertion, the TMP1962F10AXBG does not respond to $\overline{BUSRQ}$ until the wait state ends.

Note 2: This broken line indicates that output buffers are disabled, not that the signals are at indeterminate states. The pin holds the last logic value present at that pin before the bus is relinquished. This is dynamically accomplished through external load capacitances. The equipment manufacturer may maintain the bus at a predefined state by means of off-chip restores, but he or she should design, considering the time (determined by the CR constant) it takes for a signal to reach a desired state. The on-chip, integrated programmable pullup/pulldown resistors remain active, depending on internal signal states.

## 4.17 KWUP Input

Pull-up Register Active

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Low pulse width for KEY0-D | $tky_{TBL}$ | 100 | | 100 | | ns |
| High pulse width for KEY0-D | $tky_{TBH}$ | 100 | | 100 | | ns |

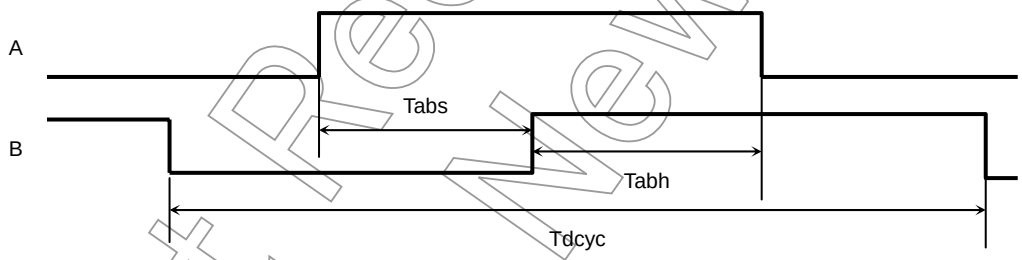
Pull-up Register Inactive

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Low pulse width for KEY0-D | $tky_{TBL}$ | 100 | | 100 | | ns |

## 4.18 Dual Pulse Input

| Parameter | Symbol | Equation | | 40.5MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| Dual input pulse period | Tdcyc | 8Y | | 395 | | ns |
| Dual input pulse setup | Tabs | Y + 20 | | 70 | | ns |
| Dual input pulse hold | Tabh | Y + 20 | | 70 | | ns |

Y: Sampling clock (fsys/2)




## 4.19 ADTRG Input

| Parameter | Symbol | Equation | | 40.5 MHz | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| ADRG low level pulse width | $tad_L$ | fsysy/2 + 20 | | 32.4 | | ns |
| ADTRG high level pulse interval | Tadh | fsysy/2 + 20 | | 32.4 | | ns |

## 5. Others

ESD

| | |
|---|---|
| MM | ±200V |
| HBM | ±1200V |

# 6.Package

P-FBGA281-1313-0.65B6