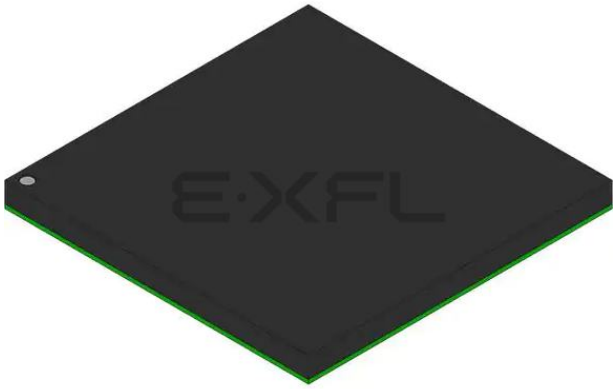

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PowerPC
Core Size	32-Bit Single-Core
Speed	56MHz
Connectivity	CANbus, EBI/EMI, SCI, SPI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	16
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 2.7V
Data Converters	A/D 32x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	388-BBGA
Supplier Device Package	388-PBGA (27x27)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mpc563mvr56

Figures

Note that exceptions can occur while an exception handler routine is executing, and multiple exceptions can become nested. It is up to the exception handler to save the appropriate machine state if it is desired that control be returned to the excepting program.

In many cases, after the exception handler handles an exception, there is an attempt to execute the instruction that caused the exception. Instruction execution continues until the next exception condition is encountered. This method of recognizing and handling exception conditions sequentially guarantees that the machine state is recoverable and processing can resume without losing instruction results.

To prevent the loss of state information, exception handlers must save the information stored in SRR0 and SRR1 soon after the exception is taken to prevent this information from being lost due to another exception being taken.

3.11.1 Exception Classes

The RCPU exception classes are shown in Table 3-18.

Table 3-18. RCPU Exception Classes

Class	Exception Type
Asynchronous, unordered	Machine check System reset
Asynchronous, ordered	External interrupt Decrementer
Synchronous (ordered, precise)	Instruction-caused exceptions

3.11.2 Ordered Exceptions

In the RCPU, all exceptions except for reset, debug port non-maskable interrupts, and machine check exceptions are ordered. Ordered exceptions satisfy the following criteria:

- Only one exception is reported at a time. If, for example, a single instruction encounters multiple exception conditions, those conditions are encountered sequentially. After the exception handler handles an exception, instruction execution continues until the next exception condition is encountered.
- When the exception is taken, no program state is lost.

3.11.3 Unordered Exceptions

Unordered exceptions may be reported at any time and are not guaranteed to preserve program state information. The processor can never recover from a reset exception. It can recover from other unordered exceptions in most cases. However, if a debug port non-maskable interrupt or machine check exception occurs during the servicing of a previous exception, the machine state information in SRR0 and SRR1 (and, in some cases, the DAR and DSISR) may not be recoverable; the processor may be in the process of saving or restoring these registers.

To determine whether the machine state is recoverable, the RI (recoverable exception) bit in SRR1 can be read. During exception processing, the RI bit in the MSR is copied to SRR1 and then cleared. The

	MSB 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Field	IRM0[1]	LVM0	IRM1	LVM1	IRM2	LVM2	IRM3	LVM3	IRM4	LVM4	IRM5	LVM5	IRM6	LVM6	IRM7	LVM7
SRESET	0000_0000_0000_0000															
Addr	0x2F C014															

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	LSB 31
Field	—															
SRESET	0000_0000_0000_0000															

[1] IRQ0 of the SIPEND register is not affected by the setting or clearing of the IRM0 bit of the SIMASK register. IRQ0 is a non-maskable interrupt.

Figure 6-18. SIU Interrupt Mask Register (SIMASK)

6.2.2.2.5 SIU Interrupt Mask Register 2 (SIMASK2)

	MSB 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Field	IRQ0[1]	LVL0	IMB IRQ0	IMB IRQ1	IMB IRQ2	IMB IRQ3	IRQ1	LVL1	IMB IRQ4	IMB IRQ5	IMB IRQ6	IMB IRQ7	IRQ2	LVL2	IMB IRQ8	IMB IRQ9
SRESET	0000_0000_0000_0000															
Addr	0x2F C048															

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	LSB 31
Field	IMB IRQ10	IMB IRQ11	IRQ3	LVL3	IMB IRQ12	IMB IRQ13	IMB IRQ14	IMB IRQ15	IRQ4	LVL4	IMB IRQ16	IMB IRQ17	IMB IRQ18	IMB IRQ19	IRQ5	LVL5
SRESET	0000_0000_0000_0000															

[1] $\overline{\text{IRQ0}}$ of the SIPEND2 register is not affected by the setting or clearing of the IRQ0 bit of the SIMASK2 register. $\overline{\text{IRQ0}}$ is a non-maskable interrupt

Figure 6-19. SIU Interrupt Mask Register 2 (SIMASK2)

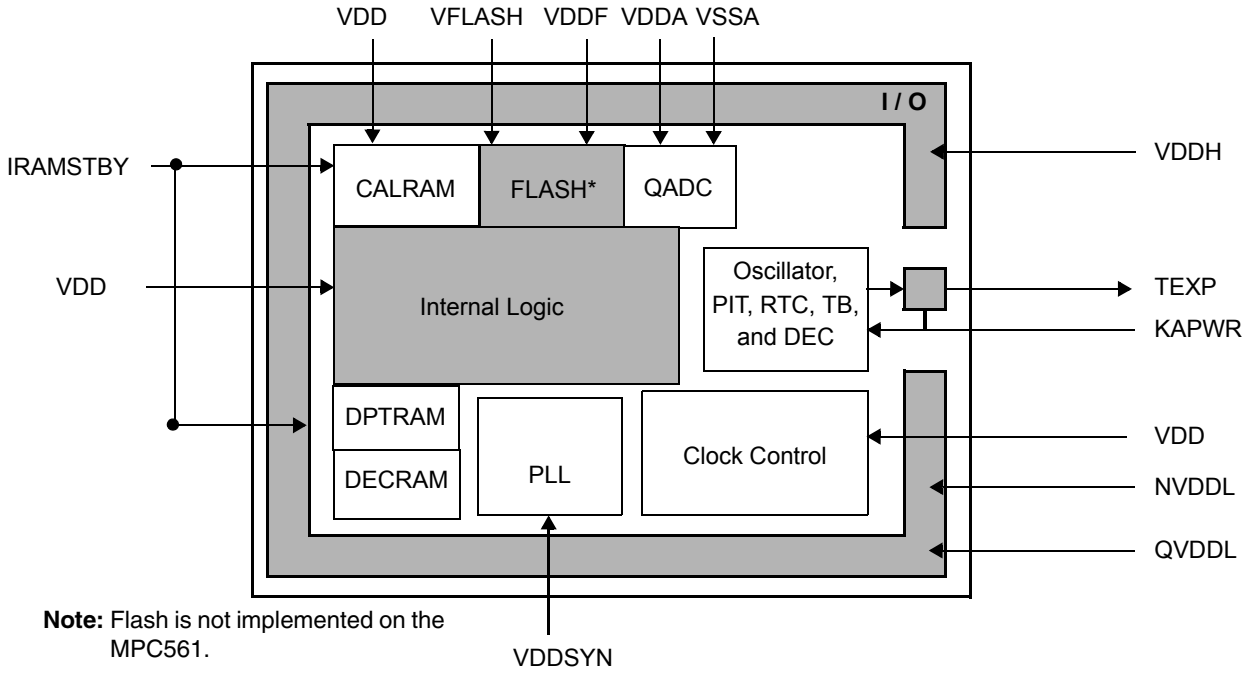

Figure 8-11. Basic Power Supply Configuration

8.8.3 Keep-Alive Power

8.8.3.1 Keep-Alive Power Configuration

Figure 8-12 is an example of a switching scheme for an optimized low-power system. SW1 and SW2 can be unified in only one switch if VDDSYN and VDD/NVDDL/QVDDL are supplied by the same source.

9.5.2 Single Beat Transfer

During the data transfer phase, the data is transferred from master to slave (in write cycles) or from slave to master (on read cycles).

During a write cycle, the master drives the data as soon as it can, but never earlier than the cycle following the address transfer phase. The master has to take into consideration the “one dead clock cycle” switching between drivers to avoid electrical contentions. The master can stop driving the data bus as soon as it samples the $\overline{TA}$ line asserted on the rising edge of the CLKOUT.

During a read cycle, the master accepts the data bus contents as valid at the rising edge of the CLKOUT in which the $\overline{TA}$ signal is sampled/asserted.

9.5.2.1 Single Beat Read Flow

The basic read cycle begins with bus arbitration, followed by the address transfer, then the data transfer. The handshakes illustrated in the following flow and timing figures (Figure 9-4, Figure 9-5, and Figure 9-6) are applicable to the fixed transaction protocol.

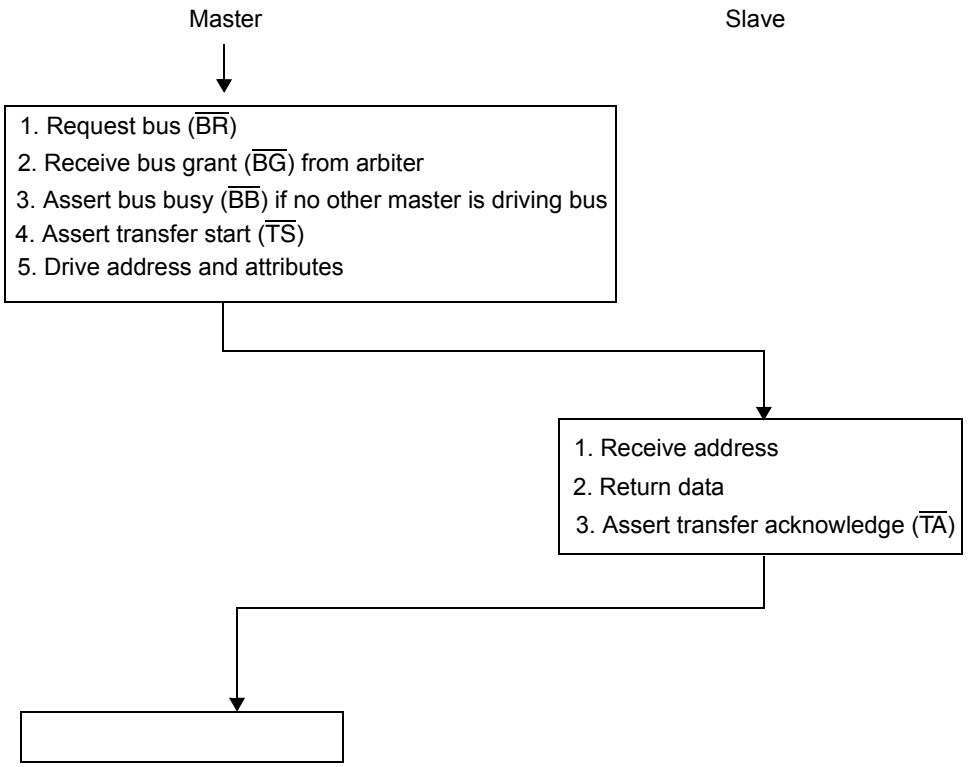

Figure 9-4. Basic Flow Diagram of a Single Beat Read Cycle

In the MPC561/MPC563, no internal master initiates write bursts. The MPC561/MPC563 is designed to perform this kind of transaction in order to support an external master that is using the memory controller services. Refer to Section 10.8, "Memory Controller External Master Support."

During the data phase of a burst-read cycle, the master receives data from the addressed slave. If the master needs more than one data beat, it asserts $\overline{\text{BDIP}}$. Upon receiving the second-to-last data beat, the master negates $\overline{\text{BDIP}}$. The slave stops driving new data after it receives the negation of the $\overline{\text{BDIP}}$ signal at the rising edge of the clock.

Burst inputs (reads) in the MPC561/MPC563 are used only for instruction cycles. Data load cycles are not supported.

Figures 9-12 through 9-21 are examples of various burst cycles, including illustrations of burst-read and burst-write cycles for both the 16- and 32-bit port sizes.

Port A signals are configured as inputs or outputs by programming the port data direction register, DDRQA. The digital input signal states are read from the port data register, PORTQA, when the port data direction register specifies that the signals are inputs. The digital data in the port data register is driven onto the port A signals when the corresponding bit in the port data direction register specifies that the signals are outputs. Refer to Appendix B, "Internal Memory Map," for more information. Since the outputs are configured as push-pull drivers, external pull-up provisions are not necessary when the output is used to drive another integrated circuit.

13.7.2 External Trigger Input Signals

The QADC64E uses two external trigger signals (ETRIG[2:1]). Each of the two input external trigger signals is associated with one of the scan queues, queue 1 or queue 2 The assignment of ETRIG[2:1] to a queue is made in the QACR0 register by the TRG bit. When TRG=0, ETRIG[1] triggers queue 1 and ETRIG[2] triggers queue 2. When TRG=1, ETRIG[1] triggers queue 2 and ETRIG[2] triggers queue 1.

NOTE

The ETRIG[2:1] pins on the MPC561/MPC563 are multiplexed with the PCS[7:6] pins.

13.7.3 Analog Power Signals

V_{DDA} and V_{SSA} signals supply power to the analog subsystems of the QADC64E module. Dedicated power is required to isolate the sensitive analog circuitry from the normal levels of noise present on the digital power supply. Refer to Appendix F, "Electrical Characteristics," for more information.

The analog supply signals (V_{DDA} and V_{SSA}) define the limits of the analog reference voltages (V_{RH} and V_{RL}) and of the analog multiplexer inputs. Figure 13-49 is a diagram of the analog input circuitry.

Table 14-14. Queue 2 Operating Modes (continued)

MQ2[3:7]	Operating Modes
01010	Interval timer single-scan mode: time = QCLK period x 2^{13}
01011	Interval timer single-scan mode: time = QCLK period x 2^{14}
01100	Interval timer single-scan mode: time = QCLK period x 2^{15}
01101	Interval timer single-scan mode: time = QCLK period x 2^{16}
01110	Interval timer single-scan mode: time = QCLK period x 2^{17}
01111	Reserved mode
10000	Reserved mode
10001	Software triggered continuous-scan mode
10010	External trigger rising edge continuous-scan mode
10011	External trigger falling edge continuous-scan mode
10100	Periodic timer continuous-scan mode: time = QCLK period x 2^{7}
10101	Periodic timer continuous-scan mode: time = QCLK period x 2^{8}
10110	Periodic timer continuous-scan mode: time = QCLK period x 2^{9}
10111	Periodic timer continuous-scan mode: time = QCLK period x 2^{10}
11000	Periodic timer continuous-scan mode: time = QCLK period x 2^{11}
11001	Periodic timer continuous-scan mode: time = QCLK period x 2^{12}
11010	Periodic timer continuous-scan mode: time = QCLK period x 2^{13}
11011	Periodic timer continuous-scan mode: time = QCLK period x 2^{14}
11100	Periodic timer continuous-scan mode: time = QCLK period x 2^{15}
11101	Periodic timer continuous-scan mode: time = QCLK period x 2^{16}
11110	Periodic timer continuous-scan mode: time = QCLK period x 2^{17}
11111	Reserved mode

NOTE

If BQ2 was assigned to the CCW that queue 1 is currently working on, then that conversion is completed before BQ2 takes effect.

Each time a CCW is read for queue 1, the CCW location is compared with the current value of the BQ2 pointer to detect a possible end-of-queue condition. For example, if BQ2 is changed to CCW3 while queue 1 is converting CCW2, queue 1 is terminated after the conversion is completed. However, if BQ2 is changed to CCW1 while queue 1 is converting CCW2, the QADC64E would not recognize a BQ2 end-of-queue condition until queue 1 execution reached CCW1 again, presumably on the next pass through the queue.

QSPI is selected it resumes storing bits in the same receive-data segment address where it left off. If more than 16 bits are transferred before negating the PCS0/$\overline{SS}$, the QSPI stores the number of bits indicated by BITS in the current receive data segment address, then increments the address and continues storing as described above.

NOTE

PCS0/$\overline{SS}$ does not necessarily have to be negated between transfers.

Once the proper number of bits (designated by BITS) are transferred, the QSPI stores the received data in the receive data segment, stores the internal working queue pointer value in CPTQP, increments the internal working queue pointer, and loads the new transmit data from the transmit data segment into the data serializer. The internal working queue pointer address is used the next time PCS0/$\overline{SS}$ is asserted, unless the CPU writes to the NEWQP first.

The DT and DSCK command control bits are not used in slave mode. As a slave, the QSPI does not drive the clock line nor the chip-select lines and, therefore, does not generate a delay.

In slave mode, the QSPI shifts out the data in the transmit data segment. The trans-mit data is loaded into the data serializer (refer to Figure 15-1) for transmission. When the PCS0/$\overline{SS}$ pin is pulled low the MISO pin becomes active and the serializer then shifts the 16 bits of data out in sequence, most significant bit first, as clocked by the incoming SCK signal. The QSPI uses CPHA and CPOL to determine which incoming SCK edge the MOSI pin uses to latch incoming data, and which edge the MISO pin uses to drive the data out.

The QSPI transmits and receives data until reaching the end of the queue (defined as a match with the address in ENDQP), regardless of whether PCS0/$\overline{SS}$ remains selected or is toggled between serial transfers. Receiving the proper number of bits causes the received data to be stored. The QSPI always transmits as many bits as it receives at each queue address, until the BITS value is reached or PCS0/$\overline{SS}$ is negated.

15.6.7 Slave Wraparound Mode

When the QSPI reaches the end of the queue, it always sets the SPIF flag, whether wraparound mode is enabled or disabled. An optional interrupt to the CPU is gen-erated when SPIF is asserted. At this point, the QSPI clears SPE and stops unless wraparound mode is enabled. A description of SPIFIE bit can be found in <XrefBlue>15.6.1.3 QSPI Control Register 2 (SPCR2).

In wraparound mode, the QSPI cycles through the queue continuously. Each time the end of the queue is reached, the SPIF flag is set. If the CPU fails to clear SPIF, it remains set, and the QSPI continues to send interrupt requests to the CPU (assuming SPIFIE is set). The user may avoid causing CPU interrupts by clearing SPIFIE.

As SPIFIE is buffered, clearing it after the SPIF flag is asserted does not immediately stop the CPU interrupts, but only prevents future interrupts from this source. To clear the current interrupt, the CPU must read QSPI register SPSR with SPIF asserted, followed by a write to SPSR with zero in SPIF (clear SPIF). Execution continues in wraparound mode even while the QSPI is requesting interrupt service from the CPU. The internal working queue pointer is incremented to the next address and the commands are

16.3.1.2 Fields for Extended Format Frames

Table 16-4 describes the message buffer fields used only for extended identifier format frames.

Table 16-4. Extended Format Frames

Field	Description
ID[28:18]/[17:15]	Contains the 14 most significant bits of the extended identifier, located in the ID_HIGH word of the message buffer.
Substitute Remote Request (SRR)	Contains a fixed recessive bit, used only in extended format. Should be set to one for Tx buffers. It will be stored as received on the CAN bus for Rx buffers.
ID Extended (IDE)	If extended format frame is used, this field should be set to one. If zero, standard format frame should be used.
ID[14:0]	Bits [14:0] of the extended identifier, located in the ID_LOW word of the message buffer.
Remote Transmission Request (RTR)	This bit is located in the least significant bit of the ID_LOW word of the message buffer 0 Data Frame 1 Remote Frame

16.3.1.3 Fields for Standard Format Frames

Table 16-5 describes the message buffer fields used only for standard identifier format frames.

Table 16-5. Standard Format Frames

Field	Description
16-bit Time Stamp	The ID_LOW word, which is not needed for standard format, is used in a standard format buffer to store the 16-bit value of the free-running timer which is captured at the beginning of the identifier field of the frame on the CAN bus.
ID[28:18]	Contains bits [28:18] of the identifier, located in the ID_HIGH word of the message buffer. The four least significant bits in this register (corresponding to the IDE bit and ID[17:15] for an extended identifier message) must all be written as logic zeros to ensure proper operation of the TouCAN.
RTR	This bit is located in the ID_HIGH word of the message buffer; 0 Data Frame 1 Remote Frame
RTR/SRR Bit Treatment	If the TouCAN transmits this bit as a one and receives it as a zero, an “arbitration loss” is indicated. If the TouCAN transmits this bit as a zero and receives it as a one, a bit error is indicated. If the TouCAN transmits a value and receives a matching response, a successful bit transmission is indicated.

16.3.1.4 Serial Message Buffers

To allow double buffering of messages, the TouCAN has two shadow buffers called serial message buffers. The TouCAN uses these two buffers for buffering both received messages and messages to be transmitted. Only one serial message buffer is active at a time, and its function depends upon the operation of the TouCAN at that time. These buffers are not accessible or visible to the user.

Table 21-6. RCW Bit Descriptions

Bits	Name	Description
0	EARB	External arbitration — Refer to Section 9.5.7, “Arbitration Phase” for a detailed description of Bus arbitration. The default value is that internal arbitration hardware is used. 0 Internal arbitration is performed 1 External arbitration is assumed
1	IP	Initial interrupt prefix — This bit defines the initial value of the MSR[IP] immediately after reset. The MSR[IP] bit defines the Interrupt Table location. 0 MSR[IP] = 0 after reset 1 MSR[IP] = 1 after reset The default value is 0. See Table 3-11 for more information.
2	BDRV	Bus pins drive strength — This bit determines the bus pins’ (address, data, and control) driving capability to be either full or reduced drive. The bus default drive strength is full; upon default, it also causes the CLKOUT drive strength to be full. See Table 6-7 for more information. BDRV controls the default state of COM[1] in the SIUMCR. 0 Full drive 1 Reduced drive
3	BDIS	Boot disable — If the BDIS bit is set, then memory controller is not activated after reset. If it is cleared then the memory controller bank 0 is active immediately after reset such that it matches any addresses. If a write to the OR0 register occurs after reset this bit definition is ignored. The default value is that the memory controller is enabled to control the boot with the $\overline{CS}$0 pin. See Section 10.7, “Global (Boot) Chip-Select Operation” for more information. 0 Memory controller bank 0 is active and matches all addresses immediately after reset 1 Memory controller is not activated after reset.
4:5	BPS	Boot port size — This field defines the port size of the boot device on reset (BR0[PS]). If a write to the OR0 register occurs after reset this field definition is ignored. See Table 10-5 and Table 10-8 for more information. 00 32-bit port (default) 01 8-bit port 10 16-bit port 11 Reserved
6:8	—	Reserved. These bits must not be high in the reset configuration word.
9:10	DBGC[0:1]	Debug pins configuration — See Section 6.2.2.1.1, “SIU Module Configuration Register (SIUMCR)” for this field definition. The default value is that these pins function as: VFLS[0:1], $\overline{BI}$, $\overline{BR}$, $\overline{BG}$ and $\overline{BB}$. See Table 6-8.
11	—	Reserved.
12	ATWC	Address type write enable configuration — The default value is that these pins function as $\overline{WE}$ pins. 0 $\overline{WE}$[0:3]/$\overline{BE}$[0:3]/AT[0:3] functions as $\overline{WE}$[0:3]/$\overline{BE}$[0:3] 1 $\overline{WE}$[0:3]/$\overline{BE}$[0:3]/AT[0:3] functions as AT[0:3] See Table 6-7.
13:14	EBDF	External bus division factor — This field defines the initial value of the external bus frequency. The default value is that CLKOUT frequency is equal to that of the internal clock (no division). See Table 8-9.
15	IWS	Interlock write select — This bit determines which interlock write operation should be used during the clear censorship operation. IWS always comes from the UC3FCFIG, it will never use the external reset configuration word (RSTCONF=0) or the default internal reset configuration word (RSTCONF=1 and $\overline{HC}$=1). 0 Interlock write is a write to any UC3F array location 1 Interlock write is a write to the UC3FMCR register.

11. Negate VSYNC
12. Return to the regular code run (issue an rfi). The first report on the VF pins is a VSYNC (VF = 011)
13. The external hardware stops sampling the program trace information upon the report on the VF pins of VSYNC

23.1.4.2 Detecting the Trace Window Start Address

When using back trace, latching the value of the status pins (VF and VFLS), and the address of the cycles marked as program trace cycle, should start immediately after the negation of reset. The start address is the first address in the program trace cycle buffer.

When using window trace, latching the value of the status pins (VF and VFLS), and the address of the cycles marked as program trace cycle, should start immediately after the first VSYNC is reported on the VF pins. The start address of the trace window should be calculated according to first two VF pins reports.

Assuming that VF1 and VF2 are the two first VF pins reports and T1 and T2 are the two addresses of the first two cycles marked with the program trace cycle attribute that were latched in the trace buffer, use the following table to calculate the trace window start address.

Table 23-4. Detecting the Trace Buffer Start Point

VF1	VF2	Starting point	Description
011 VSYNC	001 sequential	T1	VSYNC asserted followed by a sequential instruction. The start address is T1
011 VSYNC	110 branch direct taken	T1 - 4 + offset (T1 - 4)	VSYNC asserted followed by a taken direct branch. The start address is the target of the direct branch
011 VSYNC	101 branch indirect taken	T2	VSYNC asserted followed by a taken indirect branch. The start address is the target of the indirect branch

23.1.4.3 Detecting the Assertion/Negation of VSYNC

Since the VF pins are used for reporting both instruction type information and queue flush information, the external hardware must take special care when trying to detect the assertion/negation of VSYNC. When VF = 011 it is a VSYNC assertion/negation report only if the previous VF pins value was one of the following values: 000, 001, or 010.

23.1.4.4 Detecting the Trace Window End Address

The information on the status pins that describes the last fetched instruction and the last queue/history buffer flushes, changes every clock. Cycles marked as program trace cycle are generated on the external bus only when possible (when the SIU wins the arbitration over the external bus). Therefore, there is some delay between the information reported on the status pins that a cycle marked as program trace cycle will be performed on the external bus and the actual time that this cycle can be detected on the external bus.

When VSYNC is negated (through the serial interface of the development port), the CPU delays the report of the of the assertion/negation of VSYNC on the VF pins (VF = 011) until all addresses marked with the program trace cycle attribute were visible externally. Therefore, the external hardware should stop

- TP1 length=2-9
- TP2 length=2-9
- AS=0
- For alternative #1:
  - TP1 base address = base address of segment #1 vocabulary in RAM #1
  - TP2 base address = base address of segment #2 vocabulary in RAM #2
  - DS=0
- For alternative #2:
  - TP1 base address = base address of segment #2 vocabulary in RAM #1
  - TP2 base address = base address of segment #1 vocabulary in RAM #2
  - DS=1

Alternatives #1 and #2 are referred to as CLASS_2a and CLASS_2b respectively.

A.2.9.4 Left Segment Compression and Right Segment Bypass – CLASS_3

For the MPC562/MPC564, the instruction is divided into two segments. The left segment is compressed and mapped into a vocabulary. The vocabulary location is programmable. The right segment is either fully bypassed by a 16-bit field or by a shorter field which is decompressed according to fixed rules.

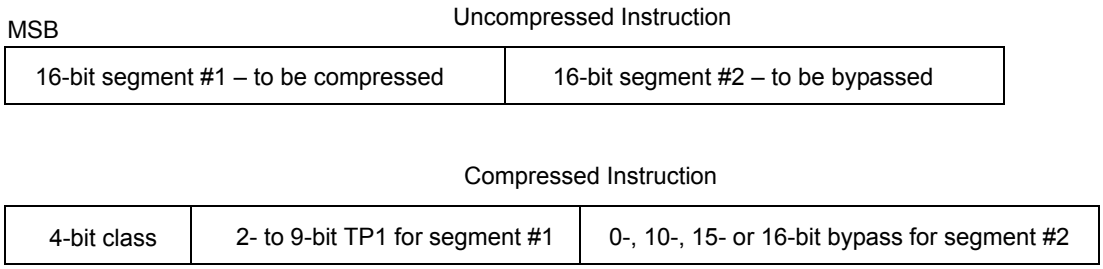

Figure A-9. CLASS_3 Instruction Layout

The definition of the class includes

- TP1 length=2-9
- TP2 length=0xB, 0xC, 0xD, or 0xE indicating a 0, 10, 15 or 16 bit bypass, respectively.
- TP1 base address = base address of segment #1 vocabulary in RAM #1, if it exists there.
- TP2 base address = base address of segment #1 vocabulary in RAM #2, if it exists there.
- DS=0
- AS=0 or 1 directing access to the vocabulary in RAM #1 or RAM #2, respectively.

When the vocabulary is located in RAM #1, the class will be referred to as CLASS_3a and when the vocabulary is located in RAM #2, the class will be referred to as CLASS_3b.

CONTROL BITS

See Table 19-24 for the PRAM Address Offset Map.

Figure D-6. FQM Parameters

D.6 Universal Asynchronous Receiver/Transmitter (UART)

The UART uses one or two TPU3 channels to provide asynchronous communications. Data word length is programmable from 1 to 14 bits. The function supports detection or generation of even, odd, and no parity. Baud rate is freely programmable and can be higher than 100 Kbaud. Eight bidirectional UART channels running in excess of 9600 baud can be implemented on the TPU3. See Freescale TPU3 Progamming Note *Asynchronous Serial Interface TPU Function (UART), (TPUPN07/D).*

Figure D-7 and Figure D-8 show all of the host interface areas for the UART function in transmitting and receiving modes.

Table F-10. Bus Operation Timing (continued)

Note: (V_{DD} = 2.6 V ± 0.1 V, V_{DDH} = 5.0 V ± 0.25 V, $T_A = T_L$ to T_H, 50 pF load unless noted otherwise)

	Characteristic	40 MHz		56 MHz[1]		Unit
		Min	Max	Min	Max	
27a	$\overline{WE}$[0:3]/$\overline{BE}$[0:3] negated to ADDR[8:31] Invalid -GPCM- write access, TRLX='0', CSNT = '1'. $\overline{CS}$ negated to ADDR[8:31] Invalid -GPCM- write access, TRLX='0', CSNT = '1', ACS = 10,ACS = ='11', EBDF = 0	8		5.71		ns
27b	$\overline{WE}$[0:3]/$\overline{BE}$[0:3] negated to ADDR[8:31] Invalid -GPCM- write access, TRLX='1', CSNT = '1'. $\overline{CS}$ negated to ADDR[8:31] Invalid -GPCM- write access, TRLX='1', CSNT = '1', ACS = 10,ACS = ='11', EBDF = 0	28		20		ns
27c	$\overline{WE}$[0:3]/$\overline{BE}$[0:3] negated to ADDR[8:31] invalid -GPCM- write access, TRLX='0', CSNT = '1'. $\overline{CS}$ negated to ADDR[8:31] Invalid -GPCM- write access, TRLX='0', CSNT = '1', ACS = 10,ACS = ='11', EBDF = 1	4		3		ns
27d	$\overline{WE}$[0:3]/$\overline{BE}$[0:3] negated to ADDR[8:31] Invalid -GPCM- write access, TRLX='1', CSNT = '1'. $\overline{CS}$ negated to ADDR[8:31] Invalid -GPCM- write access, TRLX='1', CSNT = '1', ACS = 10,ACS = ='11', EBDF = 1	24		17.25		ns
28	ADDR[8:31], TSIZ[0:1], RD/$\overline{WR}$, $\overline{BURST}$, valid to CLKOUT Rising Edge. (Slave mode Setup Time)	9		6		ns
28a	Slave Mode D[0:31] valid to CLKOUT Rising Edge	5		5		ns

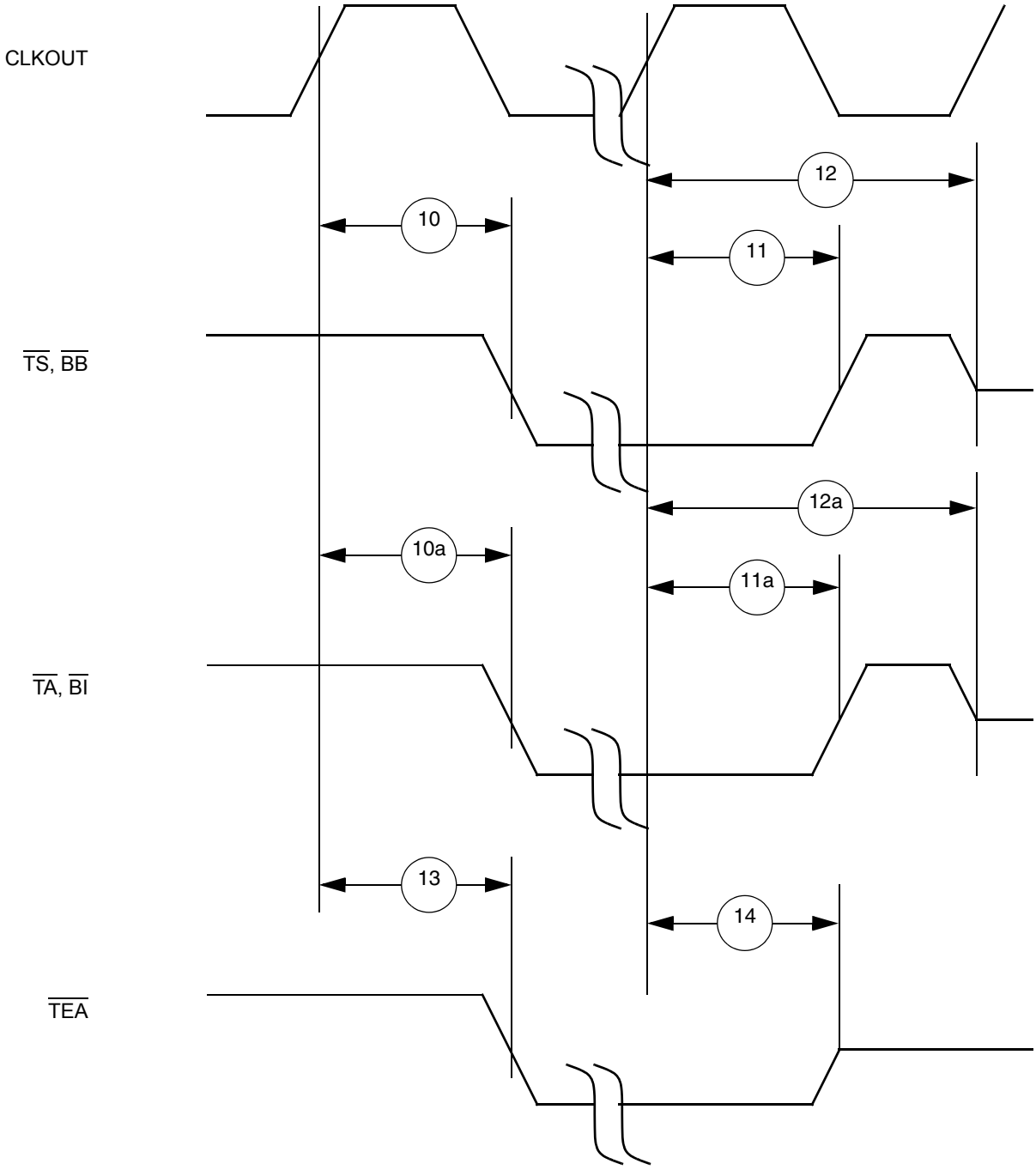

Figure F-13. Synchronous Active Pull-Up And Open Drain Outputs Signals Timing

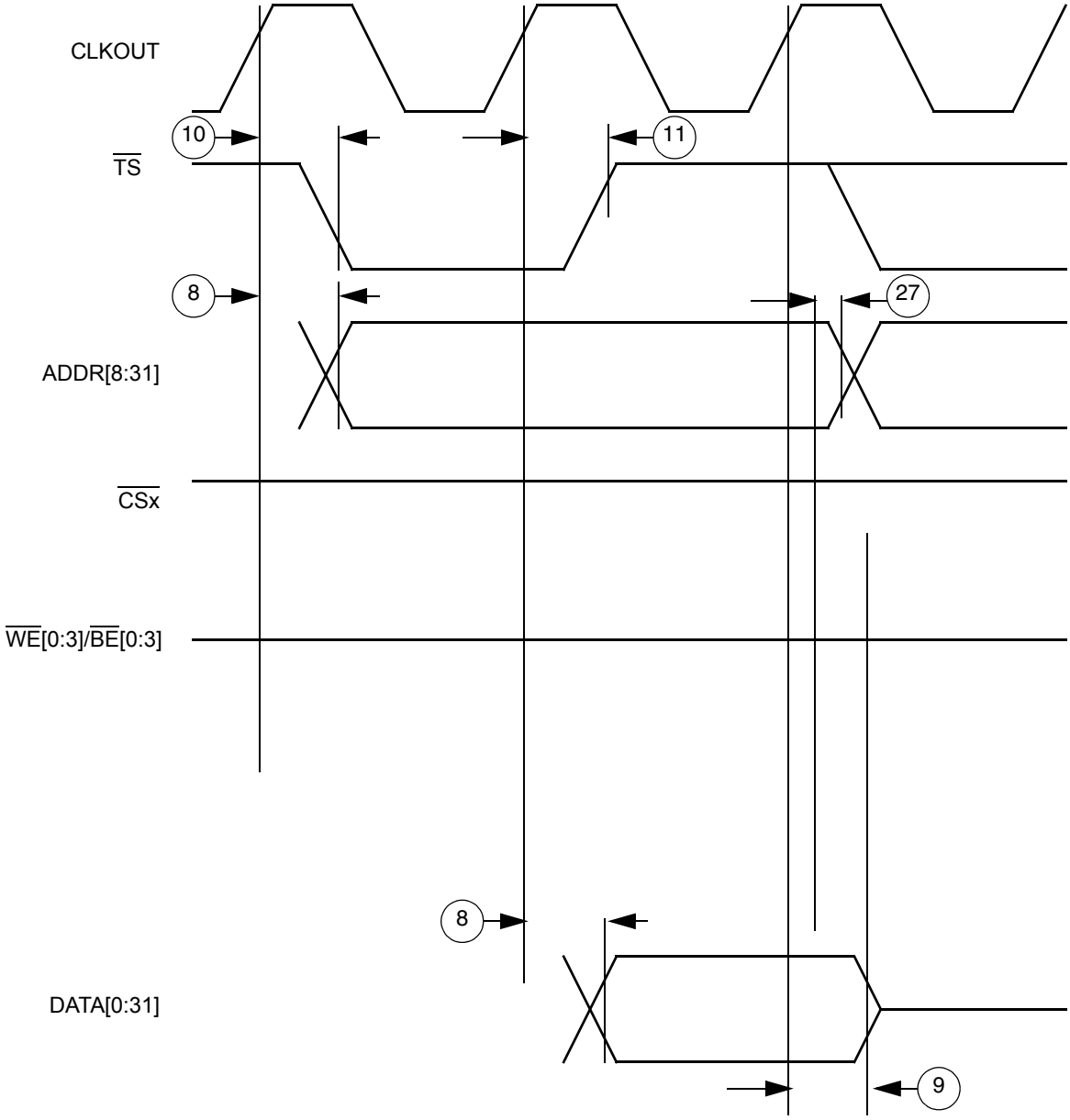

Figure F-21. Address and Data Show Cycle Bus Timing

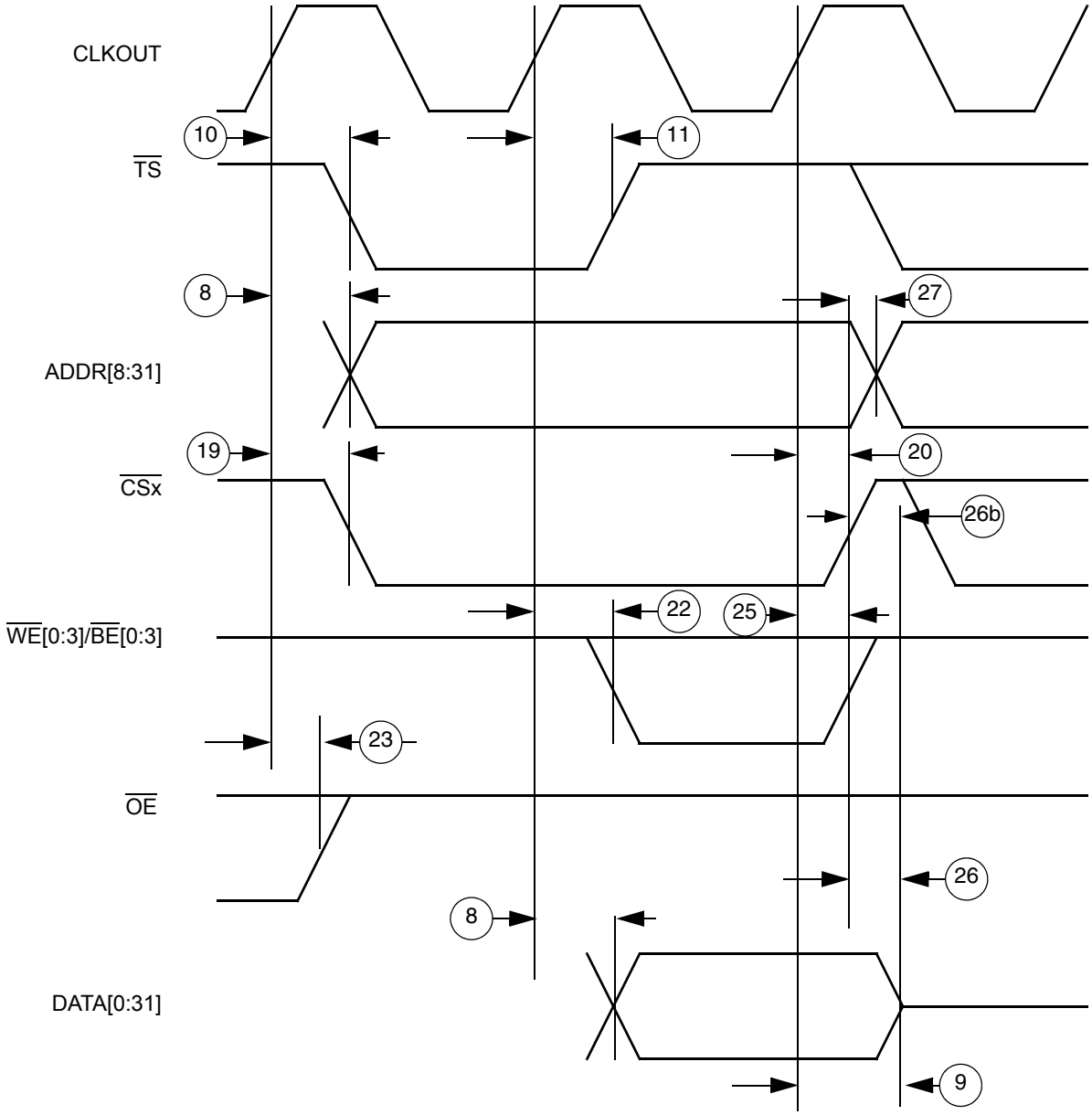

Figure F-22. External Bus Write Timing (GPCM Controlled – TRLX = '0', CSNT = '0')

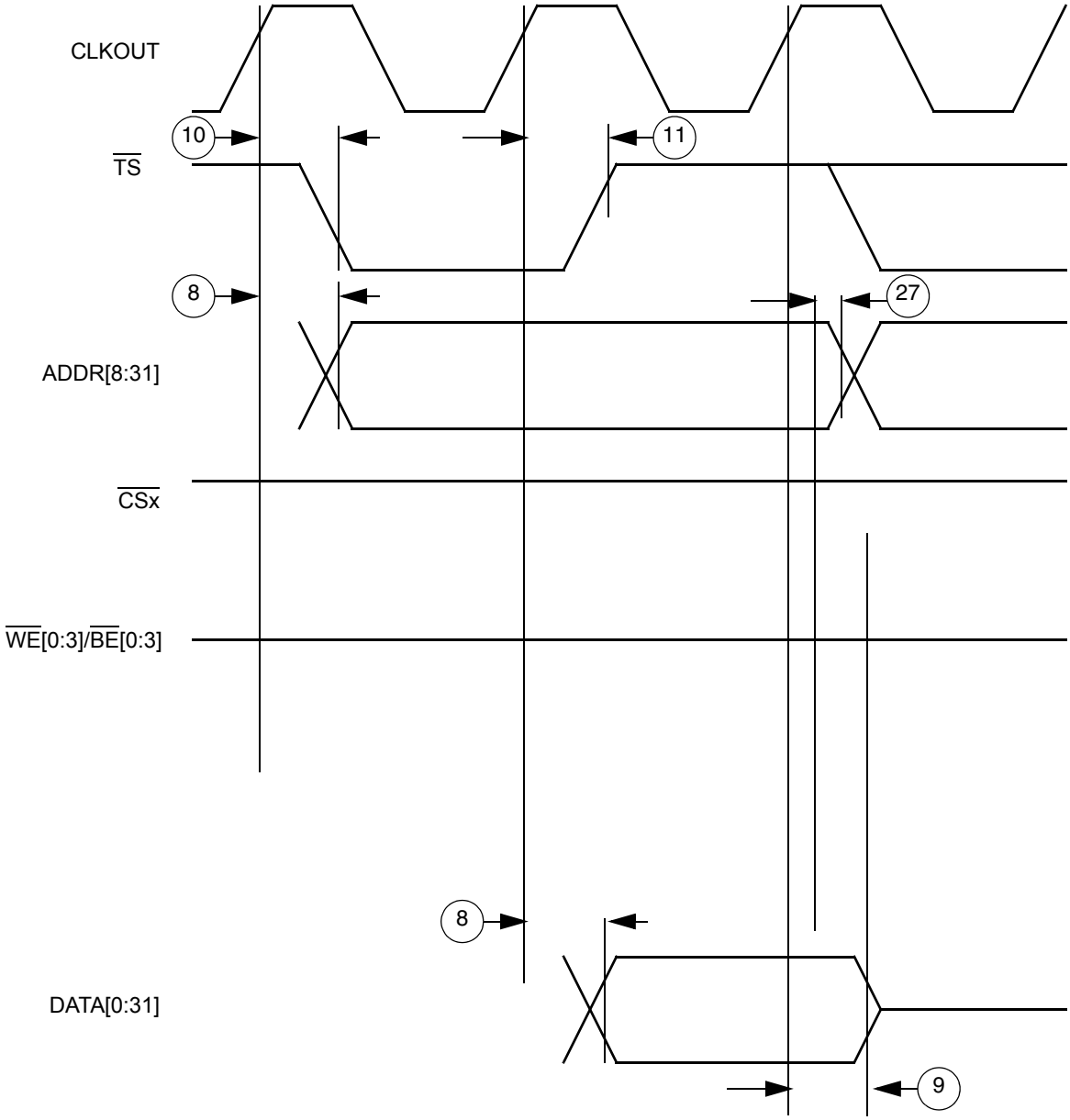

Figure G-20. Address and Data Show Cycle Bus Timing

J

K