Barrier and Eureka Synchronization

HMM-141-0

B/E Synchronization Components

The CRAY T3E system hardware contains two types of barrier and eureka (B/E) synchronization components. These components are B/E circuits and B/E contexts.

B/E Circuits

A B/E circuit performs B/E operations for all of the PEs in a system. Each PE has direct access to the 32 B/E circuits in a CRAY T3E system. Each B/E circuit receives command information from the microprocessors in the PEs and performs the appropriate B/E operation. Each B/E circuit also sends information to the microprocessors that indicates whether a B/E operation is in progress or whether it is complete.

A B/E circuit is actually composed of circuitry that is distributed among the nodes in the CRAY T3E system. Each node contains circuitry that receives and transmits B/E control signals to the PE in the node and that receives and transmits B/E control signals to all, or some, of the neighboring nodes in the interconnect network. Software defines the path that B/E control signals follow when traveling through the interconnect network.

Software also defines a root node. The root node is a node that receives B/E control signals from all of the nodes and is the node that transmits B/E control signals back to all of the nodes.

As a simplified example, all of the PEs in the system may be part of a single barrier synchronization operation. When each microprocessor reaches a barrier, the microprocessor sends a command to the B/E circuit. This command signals the B/E circuit that the microprocessor encountered a barrier and signals the B/E circuit to set an interrupt for the microprocessor when all the microprocessors in the system have encountered the barrier.

As soon as all the microprocessors have signaled the B/E circuit that they encountered a barrier, control signals from the circuitry in each node propagate through the interconnect network to the root node. The root node then sends control signals back to the circuitry in all of the nodes. These control signals indicate that a barrier event occurred.

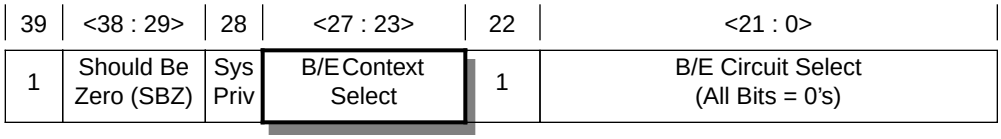
After receiving the barrier event occurred signal, the circuitry in each node sets an interrupt for the microprocessor. This interrupt indicates to the microprocessor that the barrier synchronization operation is complete.

B/E Contexts

A B/E context is a set of B/E circuits; however, in the CRAY T3E system, each B/E context contains only one B/E circuit.

The microprocessor in a PE references each B/E context by setting bits <27 : 23> of the B/E command space address to the number of the B/E context (refer to Figure 1). For example, to reference B/E context 3, the microprocessor sets bits <27 : 23> to 00011.

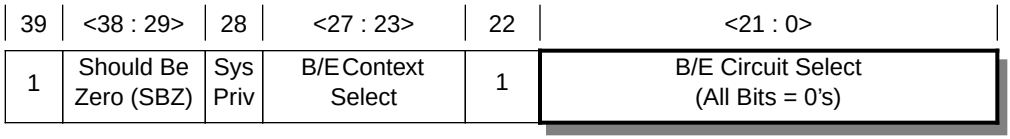
Figure 1. B/E Context Select Field



Software provides access to a B/E context to all of the PEs in the system or divides a B/E context into smaller segments and provides access to each segment to only a subset of the PEs in the system. When software divides a B/E context into smaller segments, each segment is called a B/E partition. Software may divide a B/E context into several B/E partitions. More information on B/E partitions is provided later in this document.

The microprocessor in a PE references a B/E circuit in a B/E context by setting bits <21 : 0> of the B/E command space address to the appropriate value (refer to Figure 2). In the CRAY T3E system, the B/E circuit select field is set to 0's.

Figure 2. B/E Circuit Select Field



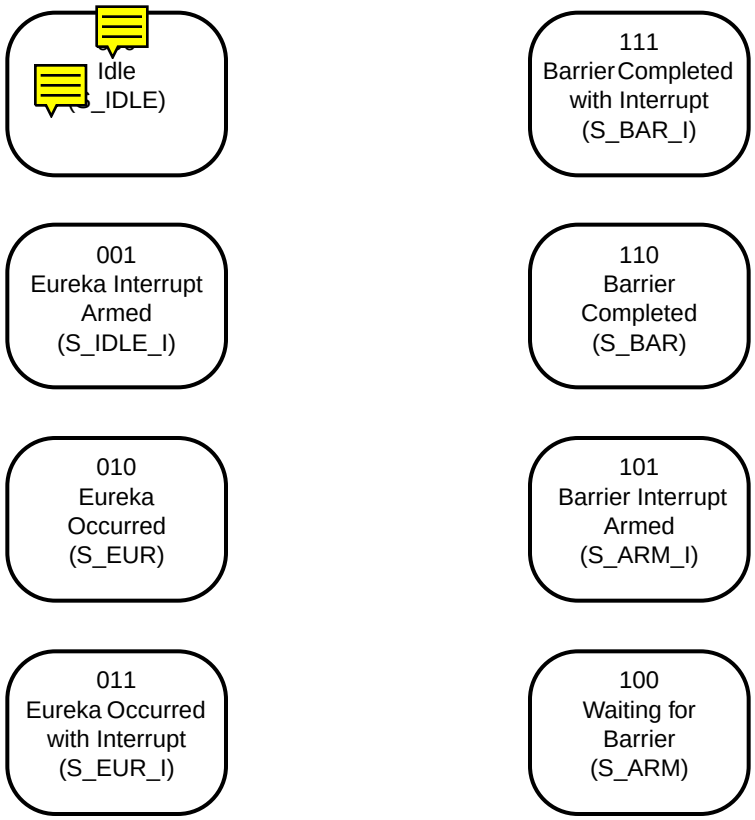
B/E Circuit States

Each B/E circuit is always in one of eight possible states. The B/E circuit states indicate that the B/E circuit is idle, that the B/E circuit is waiting for a barrier or eureka event to complete, or that a barrier or eureka event has completed.

Identifying the B/E Circuit State

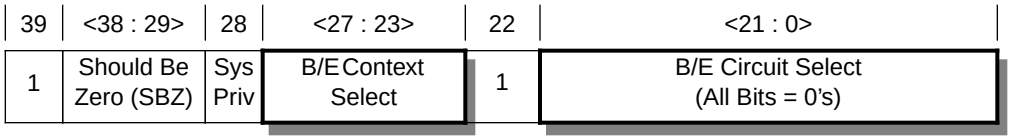
Figure 3 shows the eight B/E circuit states. The 3-bit numbers identify each state.

Figure 3. B/E Circuit States



To identify which state a B/E circuit is in, software reads information from an address in B/E command space (refer to Figure 4). When generating this address, software sets the B/E context select field of the address to indicate the context in which the requested B/E circuit resides. Software also sets the B/E circuit select field of the address to indicate the requested B/E circuit.

Figure 4. B/E Command Space Address



After receiving a read-related command and a B/E command space address from the microprocessor, the support circuitry determines the state of the B/E circuit identified in the address. The support circuitry then sends a state identification number to the microprocessor over the microprocessor data bus pins (refer to Figure 5).

Figure 5. State Identification Information on Microprocessor Data Bus Pins

The 3-bit state identification field on the microprocessor data bus pins corresponds to the 3-bit state identification numbers shown in Figure 3.

Changing the B/E Circuit State

The B/E circuit changes from one state to another when signaled by hardware or software. Hardware signals the B/E circuit to change state when a barrier or eureka event occurs. Software signals the B/E circuit to change state when software initiates a barrier or eureka synchronization operation.

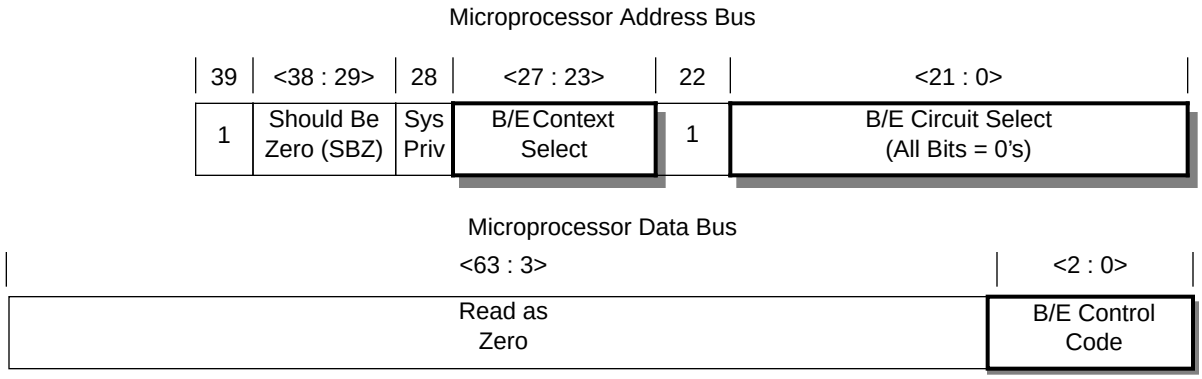
Software uses a 3-bit control code to change the state of a B/E circuit. Table 1 lists the B/E control codes and describes each code.

Table 1. B/E Control Codes

B/E Control Code	Mnemonic	Description
000	OP_CLEAR	Clear
001	Not applicable	Reserved
010	OP_EUR	Send eureka
011	OP_INT	Arm eureka interrupt
100	OP_BAR	Wait for barrier event
101	OP_BAR_I	Arm barrier interrupt
110	OP_EUR_B	Send eureka and/or wait for barrier event
111	OP_RESET	Reset

To send a control code to a B/E circuit, software writes the B/E control code to an address in B/E command space (refer to Figure 6). When generating this address, software sets the B/E context select field of the address to indicate the context in which the requested B/E circuit resides. Software sets the B/E circuit select field of the address to indicate the requested B/E circuit. Software also places the B/E control code on bits <2 : 0> of the microprocessor data bus pins.

Figure 6. B/E Control Code and B/E Command Space Address



After receiving a write-related command, a B/E command space address, and a B/E control code from the microprocessor, the support circuitry signals the B/E circuit identified in the address to change state.

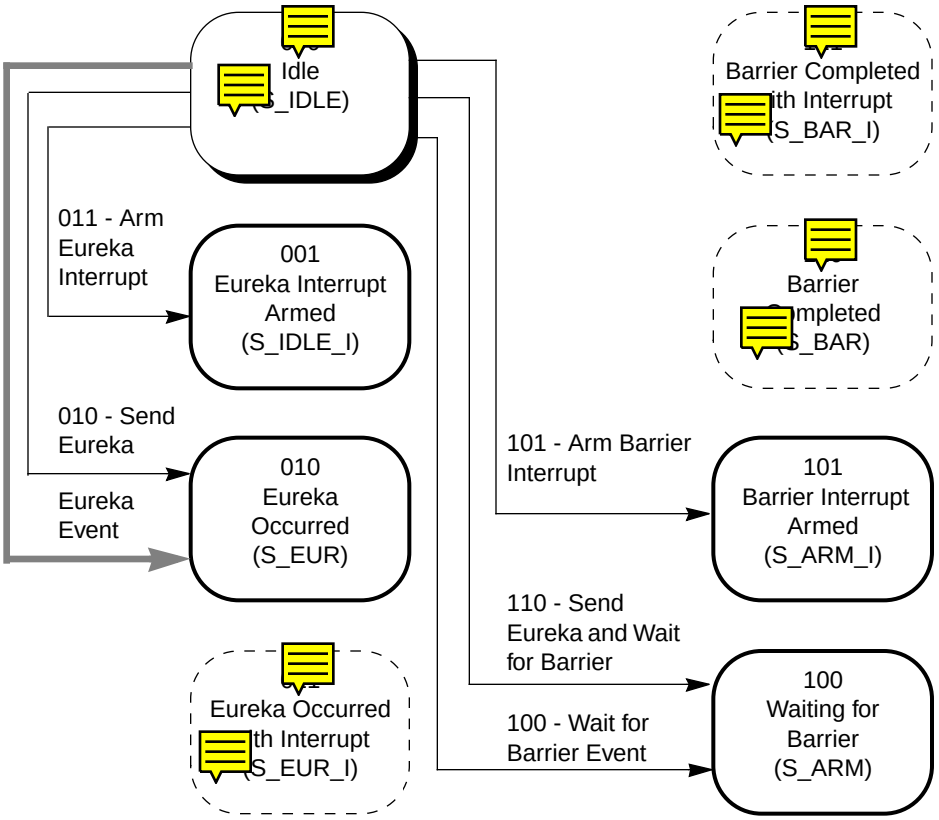
Description of Each B/E Circuit State

The following subsections describe the eight B/E circuit states.

Idle

The Idle state (S_IDLE) indicates that the B/E circuit is not performing any B/E functions. Figure 7 shows the states that are accessible from the Idle state.

Figure 7. Idle



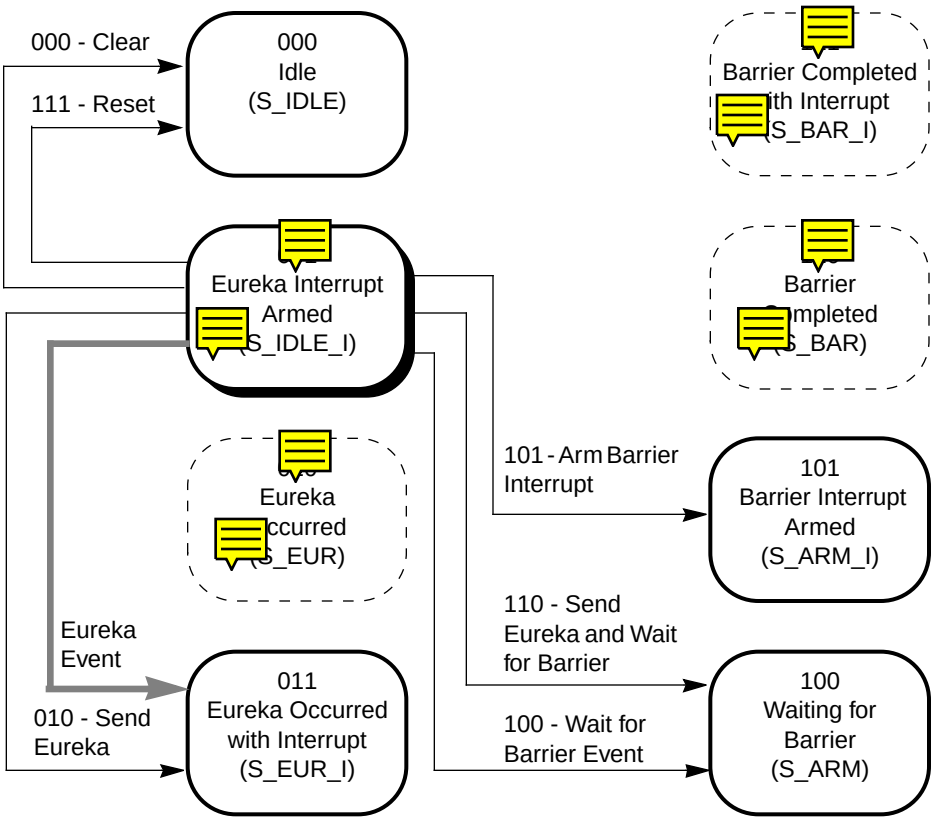
When a eureka event completes while the B/E circuit is in the Idle state, the B/E circuit changes to the Eureka Occurred state.

All other changes from the Idle state occur when software writes a B/E control code of 011, 010, 100, 101, or 110 to the B/E circuit. If software writes any other B/E control code to the B/E circuit while the circuit is in the Idle state, no operation is performed and a state change does not occur.

Eureka Interrupt Armed

The Eureka Interrupt Armed state (S_IDLE_I) indicates that the B/E circuit is waiting for a eureka event to occur and will set an interrupt when the event completes. Figure 8 shows the states that are accessible from the Eureka Interrupt Armed state.

Figure 8. Eureka Interrupt Armed



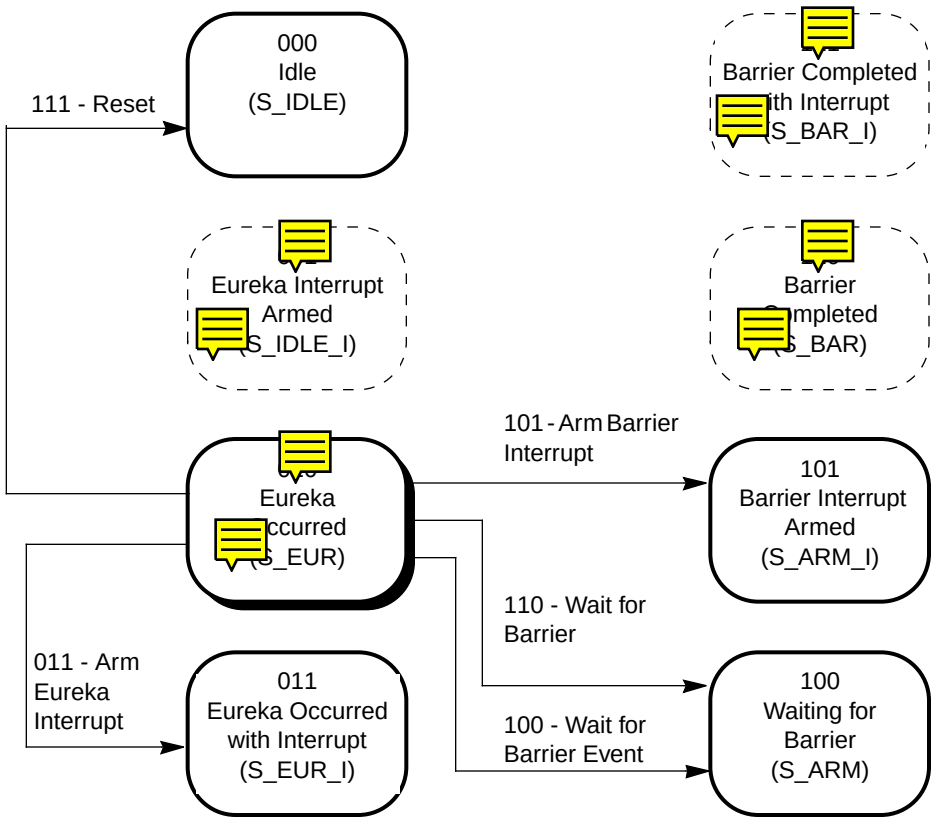
When a eureka event occurs while the B/E circuit is in the Eureka Interrupt Armed state, the B/E circuit changes to the Eureka Occurred with Interrupt state.

All other changes from the Eureka Interrupt Armed state occur when software writes a B/E control code of 000, 010, 100, 101, 110, or 111 to the B/E circuit. If software writes any other B/E control code to the B/E circuit while the circuit is in the Eureka Interrupt Armed state, no operation is performed and a state change does not occur.

Eureka Occurred

The Eureka Occurred state (S_EUR) indicates that a eureka event occurred; however, the B/E circuit does not set an interrupt for the microprocessor. Figure 9 shows the states that are accessible from the Eureka Occurred state.

Figure 9. Eureka Occurred



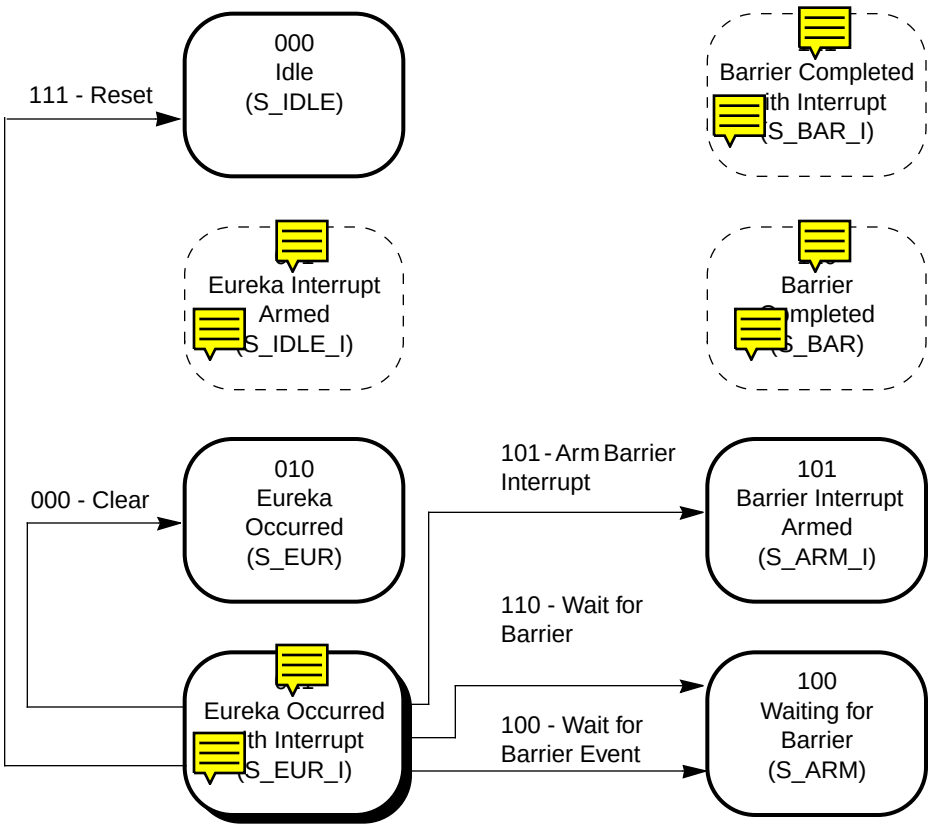
All changes from the Eureka Occurred state occur when software writes a B/E control code of 011, 100, 101, 110, or 111 to the B/E circuit. If software writes any other B/E control code to the B/E circuit while the circuit is in the Eureka Occurred state, no operation is performed and a state change does not occur.

NOTE: The 011 control code instructs the B/E circuit to arm the eureka interrupt; however, because in the Eureka Occurred state the eureka event has already occurred, the B/E circuit immediately changes to the Eureka Occurred with Interrupt state.

Eureka Occurred with Interrupt

The Eureka Occurred with Interrupt state (S_EUR_I) indicates that a eureka event occurred and that the B/E circuit has set an interrupt. Figure 10 shows the states that are accessible from the Eureka Occurred with Interrupt state.

Figure 10. Eureka Occurred with Interrupt

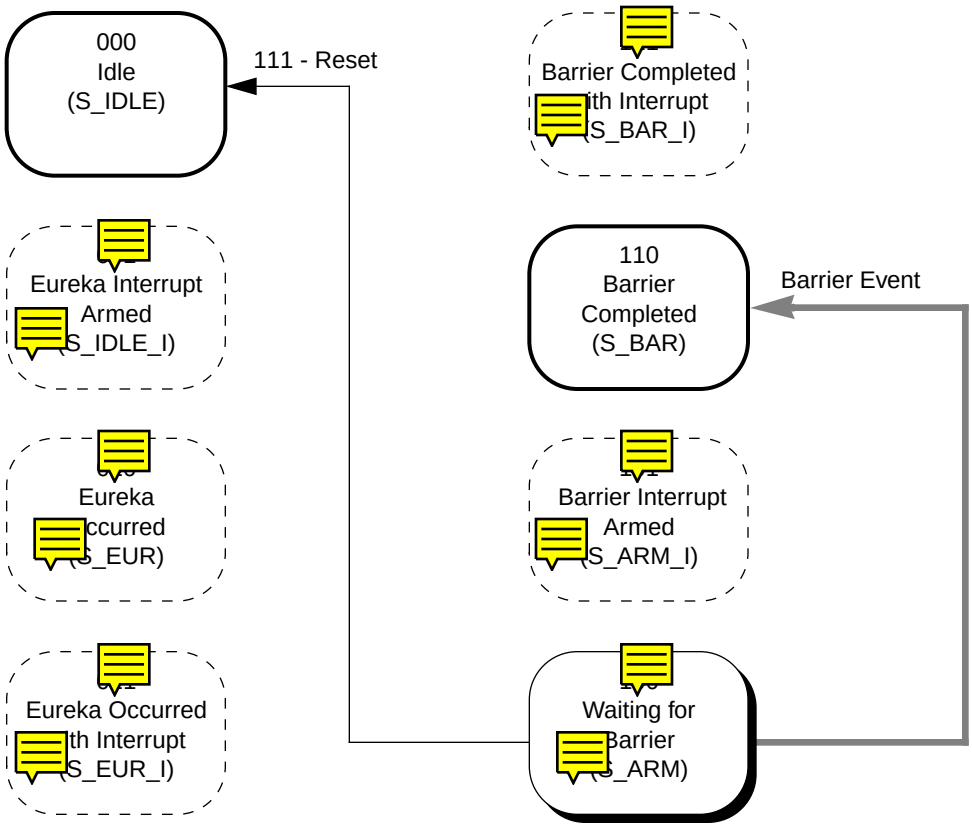


All changes from the Eureka Occurred with Interrupt state occur when software writes a B/E control code of 000, 100, 101, 110, or 111 to the B/E circuit. If software writes any other B/E control code to the B/E circuit while the circuit is in the Eureka Occurred with Interrupt state, no operation is performed and a state change does not occur.

Waiting for Barrier

The Waiting for Barrier state (S_ARM) indicates that the B/E circuit is waiting for a barrier event to complete and that the B/E circuit will not set an interrupt when the barrier synchronization operation completes. Figure 11 shows the states that are accessible from the Waiting for Barrier state.

Figure 11. Waiting for Barrier



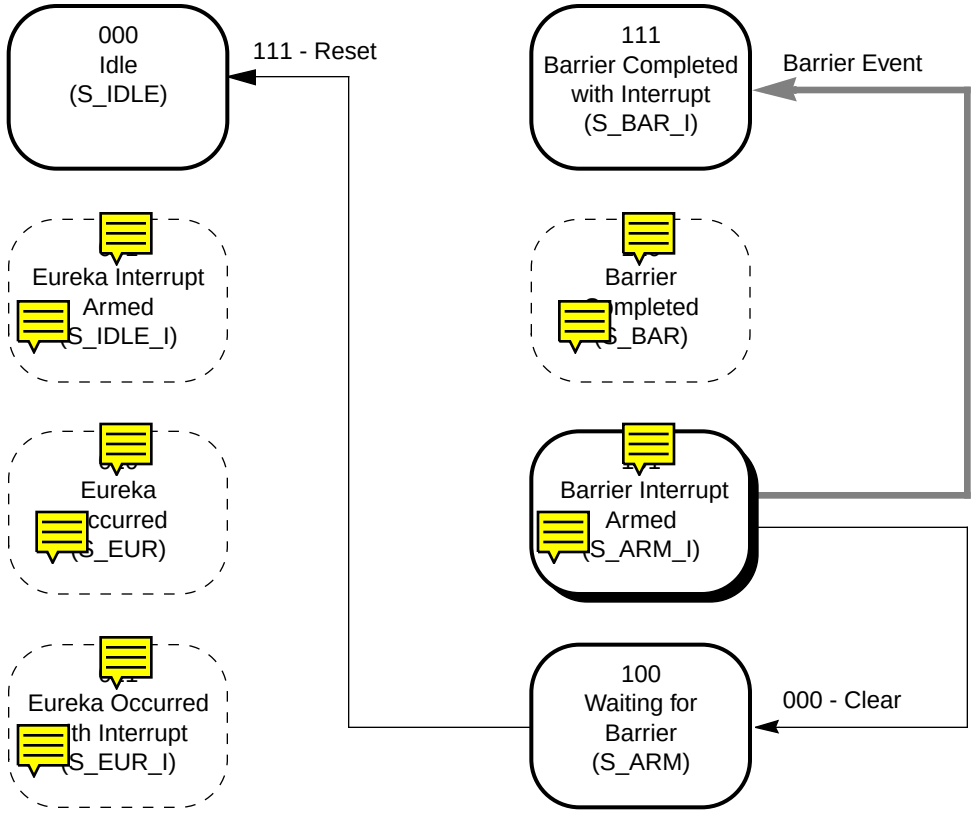
When a barrier event completes while the B/E circuit is in the Waiting for Barrier state, the B/E circuit changes into the Barrier Completed state. When software writes a B/E control code of 111 to the B/E circuit while the circuit is in the Waiting for Barrier state, the B/E circuit changes to the Idle state.

If software writes a B/E control code other than 111 to the B/E circuit while the B/E circuit is in the Waiting for Barrier state, no operation is performed and a state change does not occur.

Barrier Interrupt Armed

The Barrier Interrupt Armed state (S_ARM_I) indicates that the B/E circuit is waiting for a barrier event to complete and will set an interrupt when the barrier event completes. Figure 12 shows the states that are accessible from the Barrier Interrupt Armed state.

Figure 12. Barrier Interrupt Armed



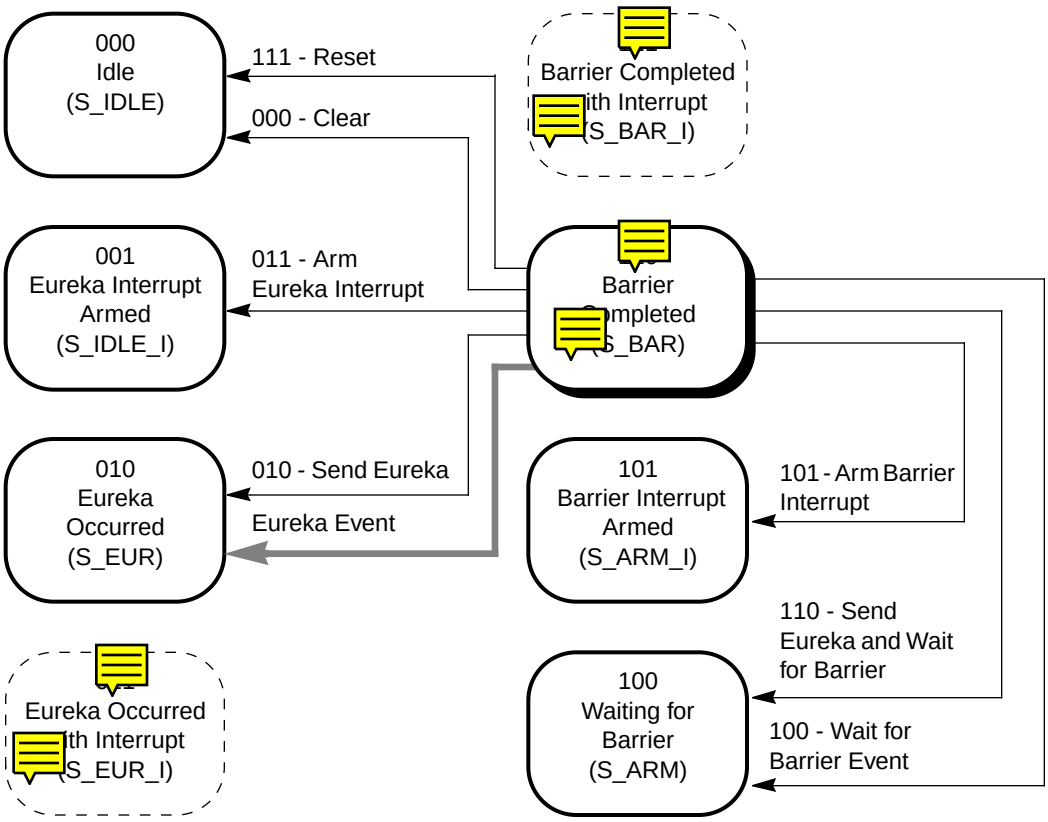
When a barrier event completes while the B/E circuit is in the Barrier Interrupt Armed state, the B/E circuit sends an interrupt to the microprocessor and then changes to the Barrier Completed with Interrupt state.

All other state changes occur when software writes a B/E control code of 000 or 111 to the B/E circuit. If software writes any other B/E control code to the B/E circuit while the circuit is in the Barrier Interrupt Armed state, no operation is performed and a state change does not occur.

Barrier Completed

The Barrier Completed state (S_BAR) indicates that a barrier event is complete and the B/E circuit did not set an interrupt. Figure 13 shows the states that are accessible from the Barrier Completed state.

Figure 13. Barrier Completed



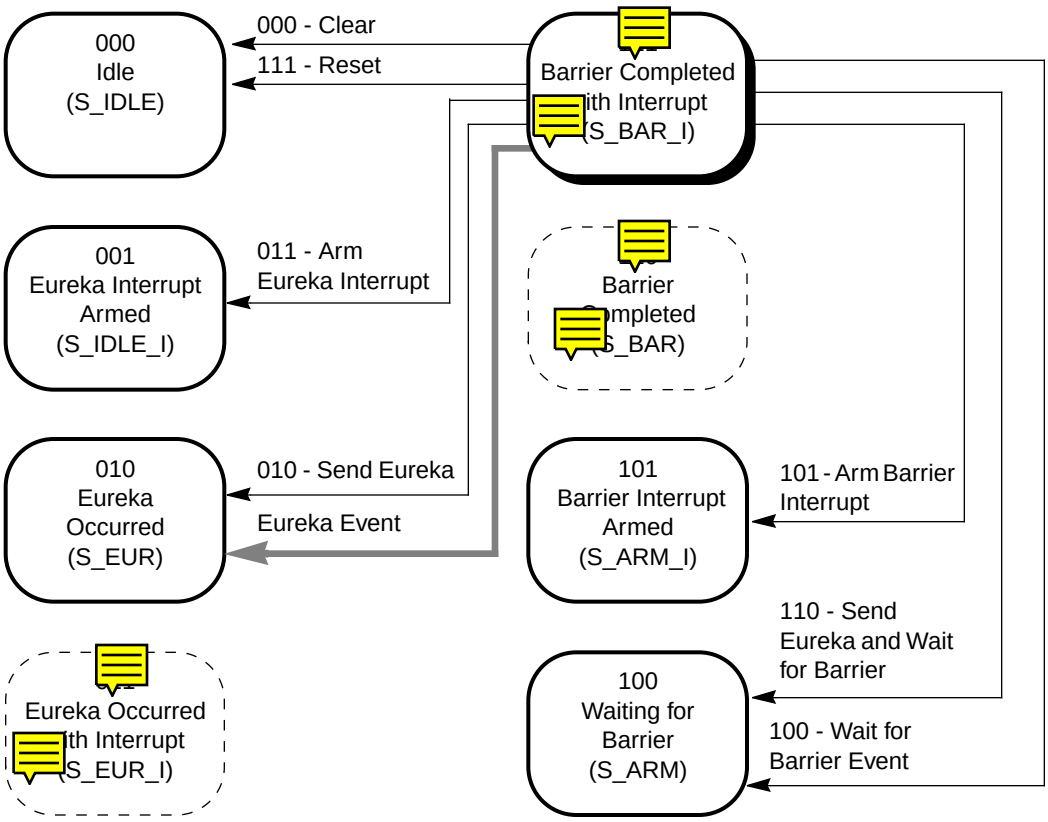
When a eureka event completes while the B/E circuit is in the Barrier Completed state, the B/E circuit changes to the Eureka Occurred state.

All other changes from the Barrier Completed state occur when software writes a B/E control code of 000, 010, 011, 100 101, 110, or 111 to the B/E circuit. If software writes any other B/E control code to the B/E circuit while the circuit is in the Barrier Completed state, no operation is performed and a state change does not occur.

Barrier Completed with Interrupt

The Barrier Completed with Interrupt state (S_BAR_I) indicates that a barrier event completed and that the B/E circuit set an interrupt. Figure 14 shows the states that are accessible from the Barrier Completed with Interrupt state.

Figure 14. Barrier Completed with Interrupt



When a eureka event completes while the B/E circuit is in the Barrier Completed with Interrupt state, the B/E circuit changes into the Eureka Occurred state.

All other changes from the Barrier Completed with Interrupt state occur when software writes a B/E control code of 000, 010, 011, 100, 101, 110, or 111 to the B/E circuit. If software writes any other B/E control code to the B/E circuit while the circuit is in the Barrier Completed with Interrupt state, no operation is performed and a state change does not occur.

Sample B/E Circuit State Transitions

The following subsections describe sample B/E circuit state transitions for these software-initiated operations:

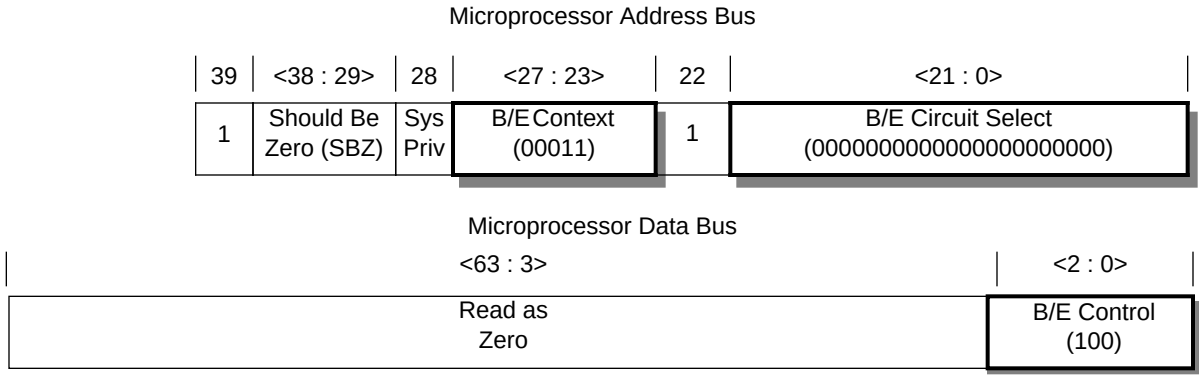
- Barrier synchronization using the polled method
- Barrier synchronization using the interrupt method
- Eureka synchronization using the interrupt method

Barrier Synchronization Using the Polled Method

When software issues a barrier synchronization operation using the polled method, the B/E circuit performs the barrier operation but does not interrupt the microprocessor when the operation is complete. The following example illustrates this method of barrier synchronization. In this example, the B/E context is context 3 and it is assigned to all of the PEs in the system. The B/E circuit is circuit 0 and it is initially in the Idle state.

1. When software running in the microprocessor reaches a barrier, the software sends a B/E control code of 100 (OP_BAR – wait for barrier event) to the B/E circuit. Figure 15 shows the format of this B/E command on the microprocessor address bus and data bus pins.

Figure 15. Sending the Wait for Barrier Event B/E Control Code



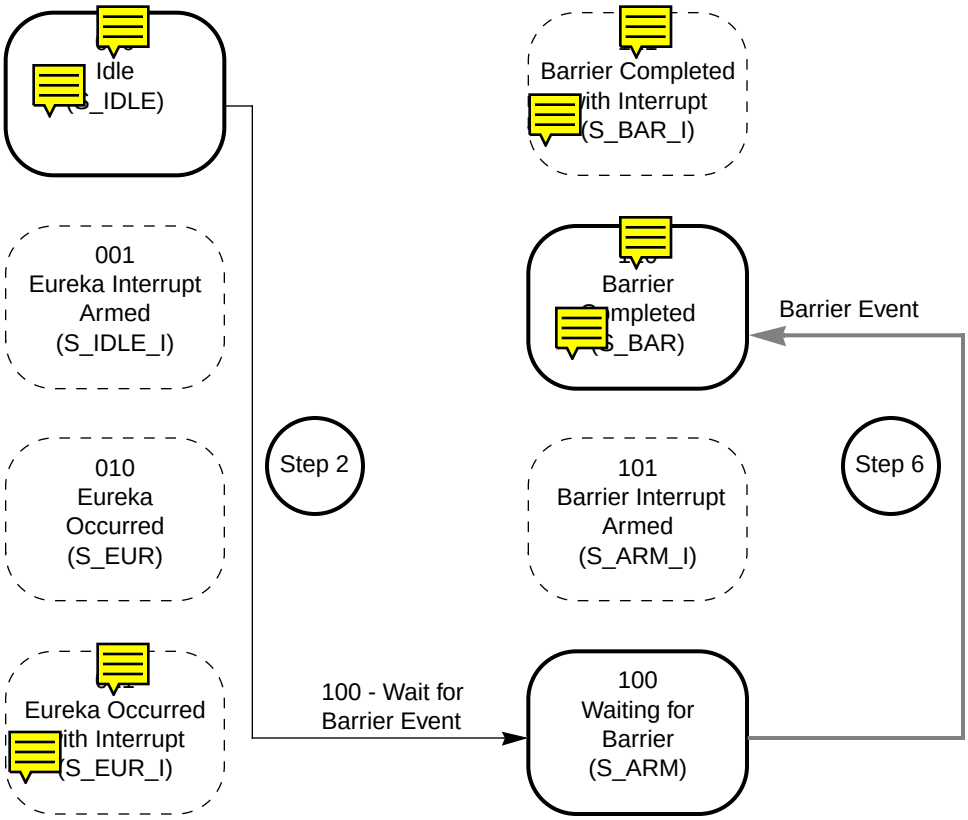
2. After receiving the B/E control code, the B/E circuit changes from the Idle state to the Waiting for Barrier state (refer to Figure 16).

3. The software then enters a loop that periodically reads the state identification number of B/E circuit 0 in B/E context 3. At this point, the state identification number is 100 (S_ARM), which indicates that the B/E circuit is waiting for a barrier event to occur.

4. Circuitry in the node sends barrier control information to the root node over the interconnect network. This control information indicates that the microprocessor encountered a barrier.

5. After receiving barrier control information from the circuitry in all of the nodes in the system, circuitry in the root node sends barrier-event-occurred control information back to the circuitry in the nodes.

Figure 16. Barrier Synchronization Using the Polled Method



6. After generating the barrier-event-occurred information, B/E circuit 0 changes state from the Waiting for Barrier state to the Barrier Completed state (refer again to Figure 16).

   NOTE: Each B/E circuit has a fixed amount of time during which all B/E operations should complete. If the B/E circuit exceeds this time limit, software may place the B/E circuit in the Idle state by sending a B/E control code of 111 (OP_RESET – reset) to the B/E circuit.

7. The software continues to periodically read the state of B/E circuit 0. At this point, the state identification number is 110 (S_BAR), which indicates that the barrier event completed.

8. After reading a state identification of 110 (S_BAR), the software running in each microprocessor continues with other program instructions.

When the B/E circuit is in the Barrier Completed state (S_BAR), software can immediately initiate a new barrier or eureka operation.

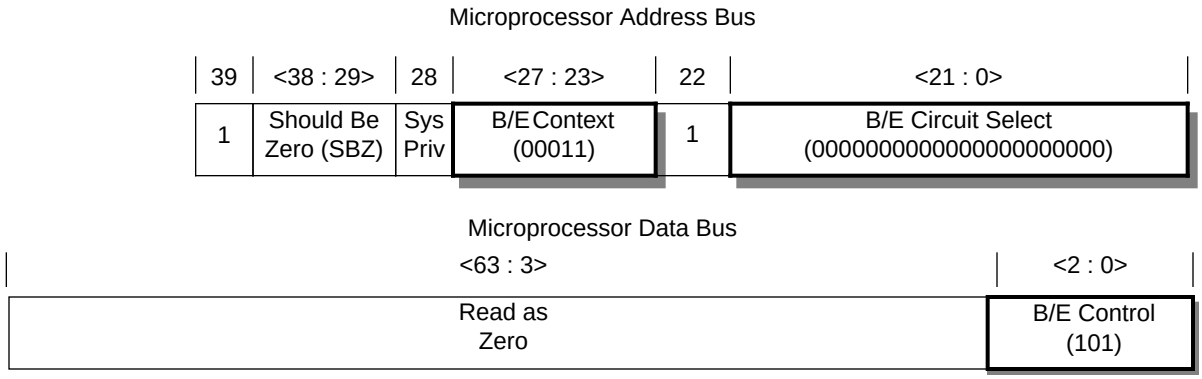
Barrier Synchronization Using the Interrupt Method

When software issues a barrier synchronization operation using the interrupt method, the B/E circuit performs the barrier operation and interrupts the microprocessor when the operation is complete. The following example illustrates this method of barrier synchronization. In this example, the B/E context is context 3 and it is assigned to all of the PEs in the system. The B/E circuit is circuit 0 and it is initially in the Idle state.

1. When the software running in the microprocessor reaches a barrier, the software enables the B/E interrupt for B/E context 3. To do this, software sets bit 3 (BAR[3]) of the interrupt enable (IR_ENABLE) register to 1.

2. Software sends a B/E control code of 101 (OP_BAR_I – arm barrier interrupt) to the B/E circuit. Figure 17 shows the format of this B/E command on the microprocessor address bus and data bus pins.

Figure 17. Sending the Arm Barrier Interrupt B/E Control Code

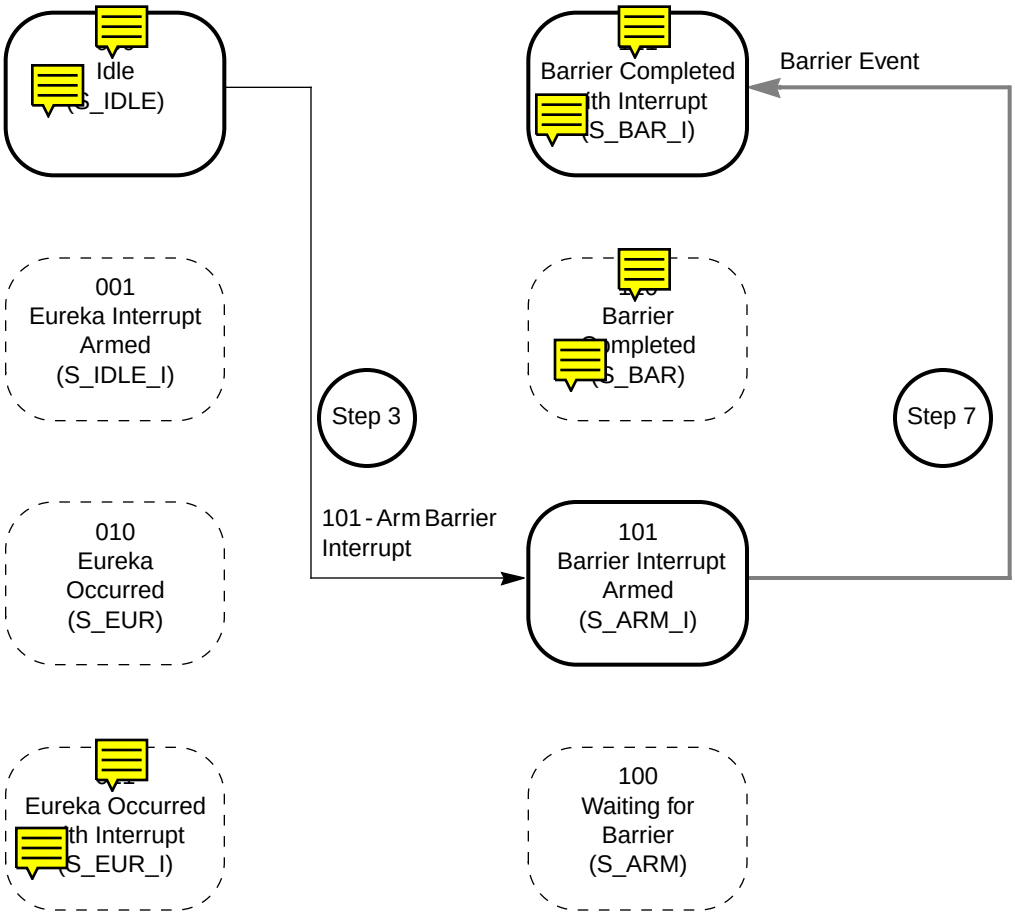


3. After receiving the B/E control code, the B/E circuit changes from the Idle state to the Barrier Interrupt Armed state (refer to Figure 18).

4. The software running in the microprocessor continues with other program instructions.

5. Circuitry in the node sends barrier control information to the root node over the interconnect network. This control information indicates that the microprocessor encountered a barrier.

Figure 18. Barrier Synchronization Using the Interrupt Method



6. After receiving barrier control information from the circuitry in all of the nodes in the system, circuitry in the root node sends barrier-event-occurred control information back to all of the nodes in the system.

7. After generating the barrier-event-occurred information, the B/E circuit changes state from the Barrier Interrupt Armed state to the Barrier Completed with Interrupt state (refer again to Figure 18).

   NOTE: Each B/E circuit has a fixed amount of time during which all B/E operations should complete. If the B/E circuit exceeds this time limit, software may place the B/E circuit into the Idle state by sending a B/E control code of 111 (OP_RESET – reset) to the B/E circuit.

8. The B/E circuit sets a B/E interrupt for the microprocessor. To do this, the B/E circuit sets bit 3 (BAR[3]) of the interrupt status (IR_STATUS) register to 1. This bit indicates that a B/E interrupt occurred for B/E context 3.

9. After receiving an interrupt, the software running in each microprocessor reads the value of the IR_STATUS register and determines that B/E context 3 set the interrupt.

10. The software running in each microprocessor reads the state identification of the B/E circuits in B/E context 3. At this point, the state identification number for B/E circuit 0 is 111 (S_BAR_I), which indicates that B/E circuit 0 is in the Barrier Completed with Interrupt state.

11. Software writes a 1 to bit 3 (BAR[3]) of the IR_STATUS register to reset the B/E interrupt for B/E context 3; it then continues with other program instructions.

When the B/E circuit is in the Barrier Completed with Interrupt state (S_BAR_I), software may immediately initiate a new barrier or eureka operation.

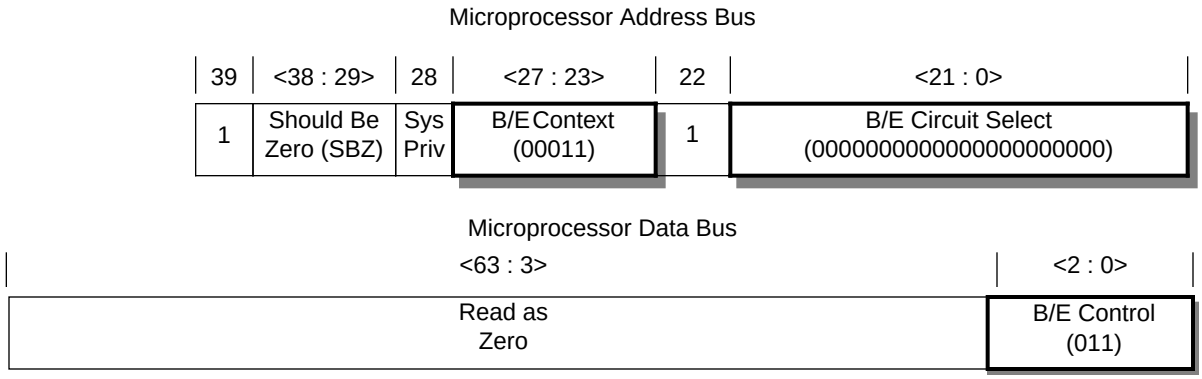
Eureka Synchronization Using the Interrupt Method

When software issues a eureka synchronization operation using the interrupt method, the B/E circuit performs the eureka operation and interrupts the microprocessor when the operation is complete. The following example illustrates this method of eureka synchronization. In this example, the B/E context is context 3 and it is assigned to all of the PEs in the system. The B/E circuit is circuit 0 and it is initially in the Idle state.

1. When the software running in the microprocessor initiates a eureka operation, the software enables the B/E interrupt for B/E context 3. To do this, software sets bit 3 (BAR[3]) of the IR_ENABLE register to 1.

2. Software sends a B/E control code of 011 (OP_INT – arm eureka interrupt) to the B/E circuit. Figure 19 shows the format of this B/E command on the microprocessor address bus and data bus pins.

Figure 19. Sending the Arm Eureka Interrupt B/E Control Code

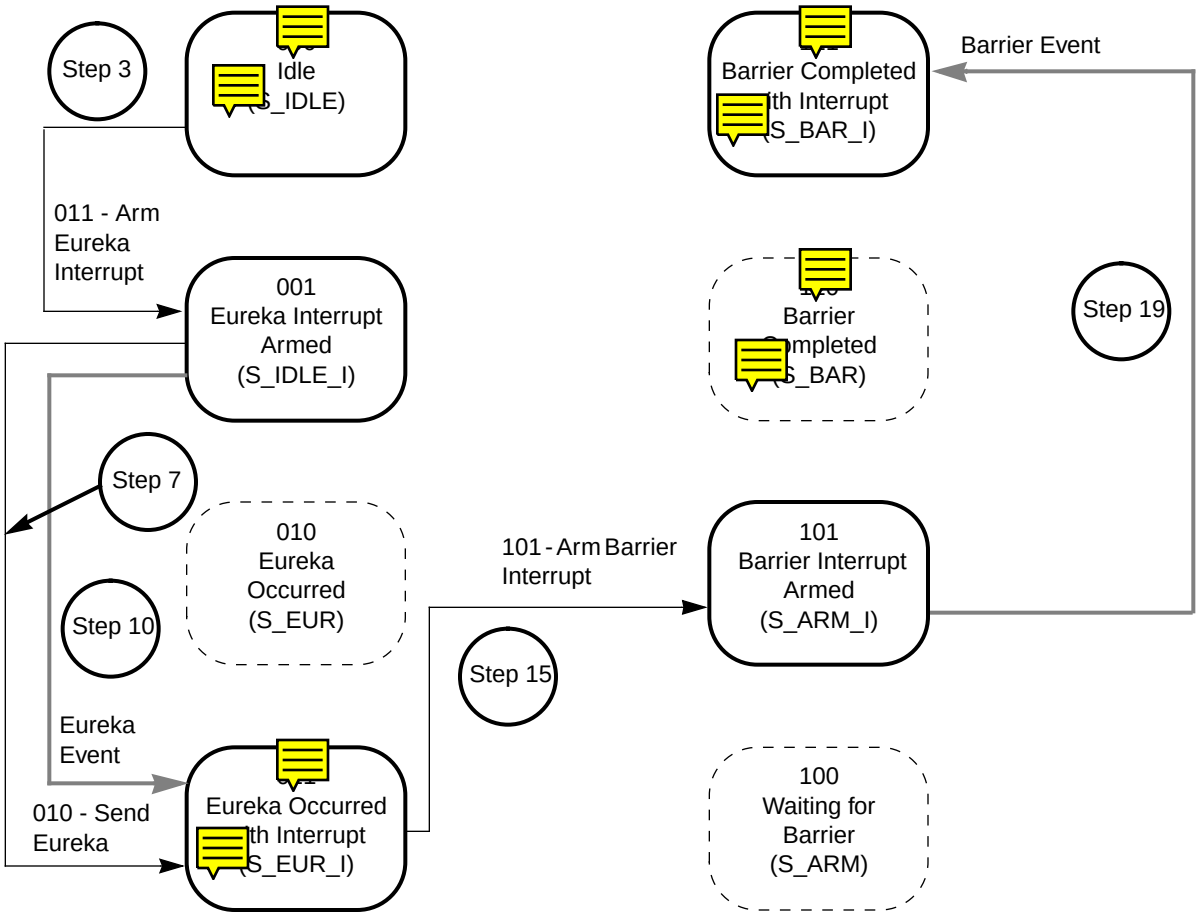


3. After receiving the B/E control code, the B/E circuit changes from the Idle state to the Eureka Interrupt Armed state (refer to Figure 20).

4. Circuitry in the node sends control information to the root node over the interconnect network. This control information indicates that the microprocessor started a eureka process.

5. The software running in the microprocessor continues with program instructions that relate to the eureka operation. For example, the software may continue searching a database.

Figure 20. Eureka Synchronization Using the Interrupt Method



6. The software running in one of the PEs (PE 1 in node 1 for this example) completes the program instructions that are related to the eureka operation. For example, the software in PE 1 finds the data for a database search. The software in PE 1 then sends a B/E control code of 010 (OP_EUR – send eureka) to the B/E circuit.

7. After receiving the B/E control code, the circuitry in node 1 changes from the Eureka Interrupt Armed state to the Eureka Occurred with Interrupt state (refer again to Figure 20).

8. Circuitry in node 1 sends control information to the root node over the interconnect network. This control information indicates that the microprocessor in PE 1 initiated a eureka event.

9. After receiving eureka-event control information from node 1, the root node sends eureka-event-occurred control information back to all of the nodes in the system.

10. After generating the eureka-event-occurred information, the B/E circuit changes state from the Eureka Interrupt Armed state into the Eureka Occurred with Interrupt state (refer again to Figure 20).

11. The B/E circuit sets a B/E interrupt for the microprocessor. To do this, the B/E circuit sets bit 3 (BAR[3]) of the IR_STATUS register to 1. This bit indicates that a B/E interrupt occurred for B/E context 3.

12. After receiving an interrupt, the software running in each microprocessor reads the value of the IR_STATUS register and determines that B/E context 3 set the interrupt.

13. The software running in each microprocessor reads the state identification of the B/E circuits in B/E context 3. At this point, the state identification number for B/E circuit 0 is 011 (S_EUR_I), which indicates that B/E circuit 0 is in the Eureka Completed with Interrupt state.

14. Software writes a 1 to bit 3 (BAR[3]) of the IR_STATUS register to reset the B/E interrupt.

15. Software sends a B/E control code of 101 (OP_BAR_I – arm barrier interrupt) to the B/E circuit.

16. After receiving the B/E control code, the B/E circuit changes from the Eureka Occurred with Interrupt state to the Barrier Interrupt Armed state (refer to Figure 20).

17. The software running in the microprocessor continues with other program instructions.

18. Circuitry in the node sends control information to the root node over the interconnect network. This control information indicates that the microprocessor encountered a barrier.

19. After receiving barrier control information from circuitry in all of the nodes, the root node sends barrier-event-occurred control information back to all of the nodes in the system.

20. After generating barrier-event-occurred information, the B/E circuit changes state from the Barrier Interrupt Armed state to the Barrier Completed with Interrupt state (refer again to Figure 20).

    NOTE: Each B/E circuit has a fixed amount of time during which all B/E operations should complete. If the B/E circuit exceeds this time limit, software may place the B/E circuit into the Idle state by sending a B/E control code of 111 (OP_RESET – reset) to the B/E circuit.

21. The B/E circuit sets a B/E interrupt for the microprocessor. To do this, the B/E circuit sets bit 3 (BAR[3]) of the IR_STATUS register to 1. This bit indicates that a B/E interrupt occurred for B/E context 3.

22. After receiving an interrupt, the software running in each microprocessor reads the value of the IR_STATUS register and determines that B/E context 3 set the interrupt.

23. The software running in each microprocessor reads the state identification of the B/E circuits in B/E context 3. At this point, the state identification number for B/E circuit 0 is 111 (S_BAR_I), which indicates that B/E circuit 0 is in the Barrier Completed with Interrupt state.

24. Software writes a 1 to bit 3 (BAR[3]) of the IR_STATUS register to reset the B/E interrupt and then continues with other program instructions.

When the B/E circuit is in the Barrier Completed with Interrupt state (S_BAR_I), software can immediately initiate a new barrier or eureka operation.

B/E Partitions

Software may divide each B/E context into B/E partitions. Software may provide access to each B/E partition to only a subset of the PEs in the system. As an example, Figure 21 shows the nodes for a CRAY T3E AC20 system. (For clarity, Figure 21 does not show the communication links that complete the torus in each dimension.) Software may divide some of the B/E contexts for this system into the B/E partitions shown in Table 2.

Figure 21. CRAY T3E AC20 System Nodes

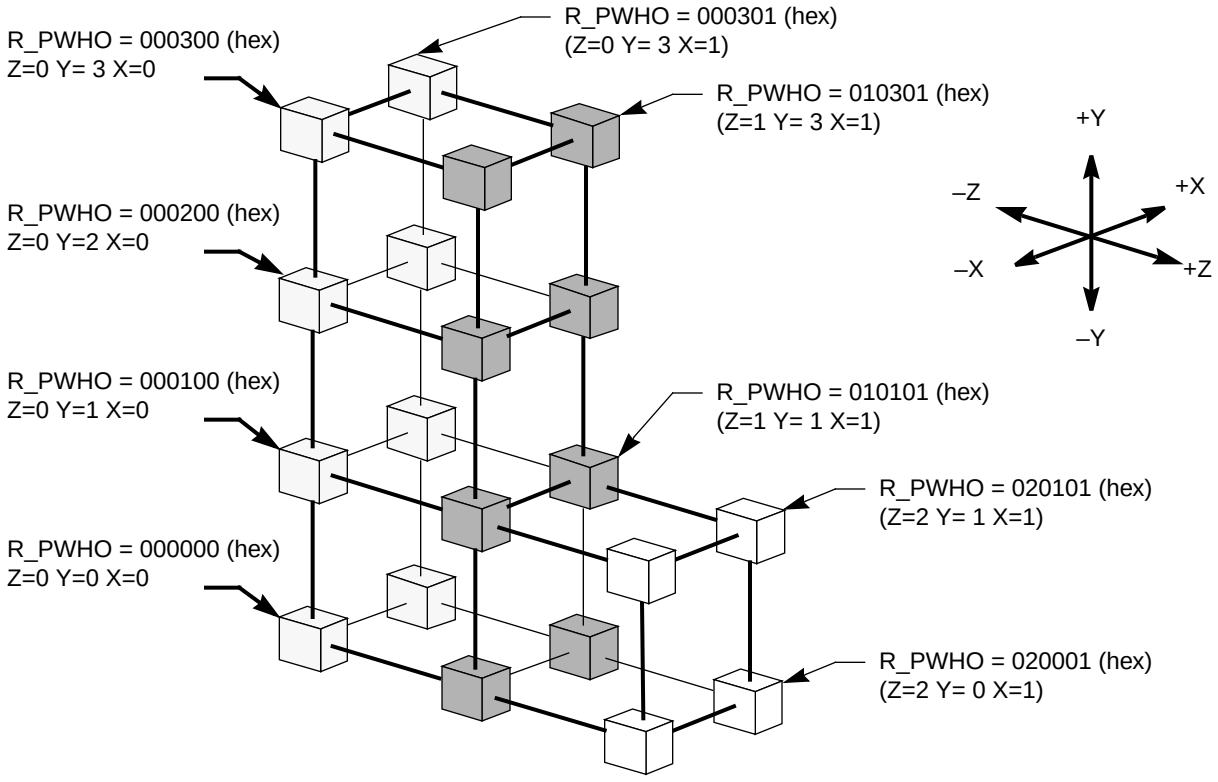


Table 2. Sample B/E Partitions

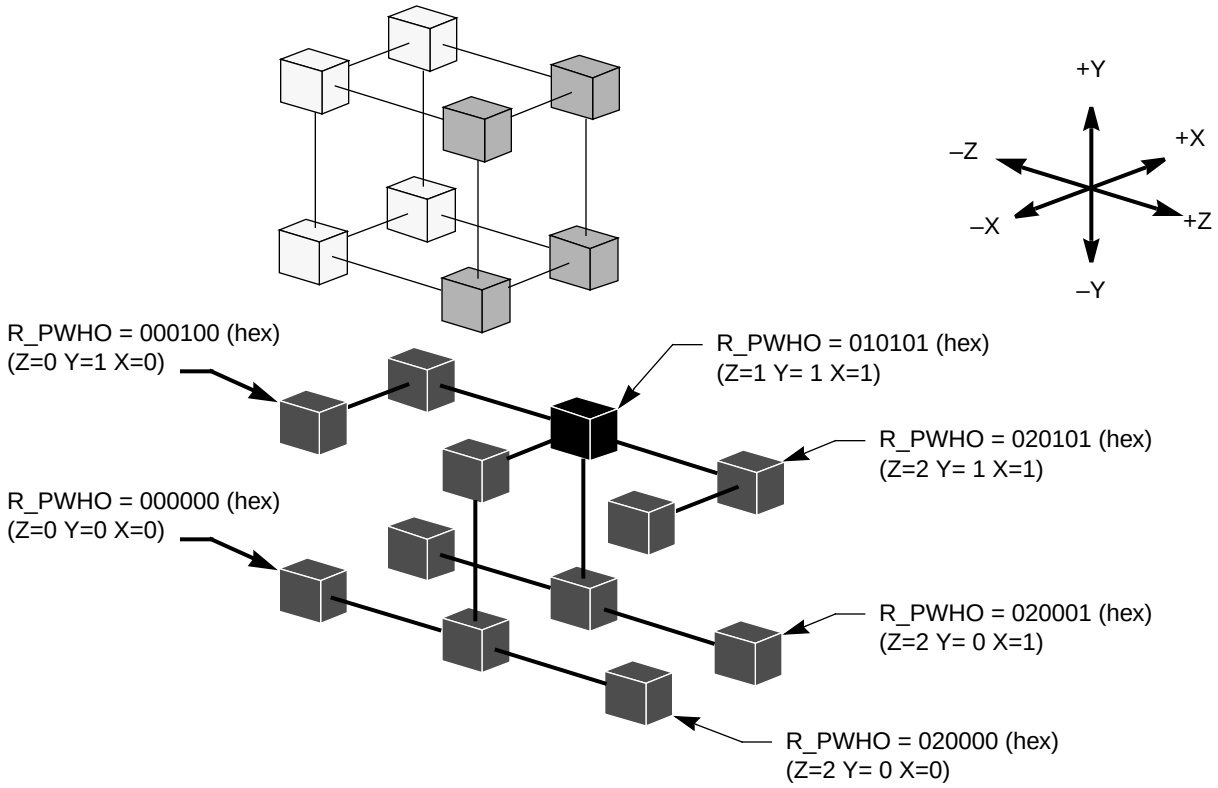
B/E Context	B/E Partition	Nodes (Hex R_PWHO Values)
0	A	All the nodes in the system.
3	A	000200, 000300, 000201, 000301 010200, 010300, 010201, 010301
3	B	000000, 000100, 000001, 000101 010000, 010100, 010001, 010101 020000, 020100, 020001, 020101

Root Node

When software defines a B/E partition, software first identifies a root node. The root node receives B/E control information from circuitry in all of the nodes in a B/E partition and is the node that sends B/E control information back to all of the nodes in a B/E partition.

As an example, Figure 22 shows B/E partition B of B/E context 3 as described in Table 2. In this example, software chose the node with a R_PWHO register of 010101(hex) (Z=1 Y=1 X=1) as the root node.

Figure 22. Root Node and B/E Communication Links



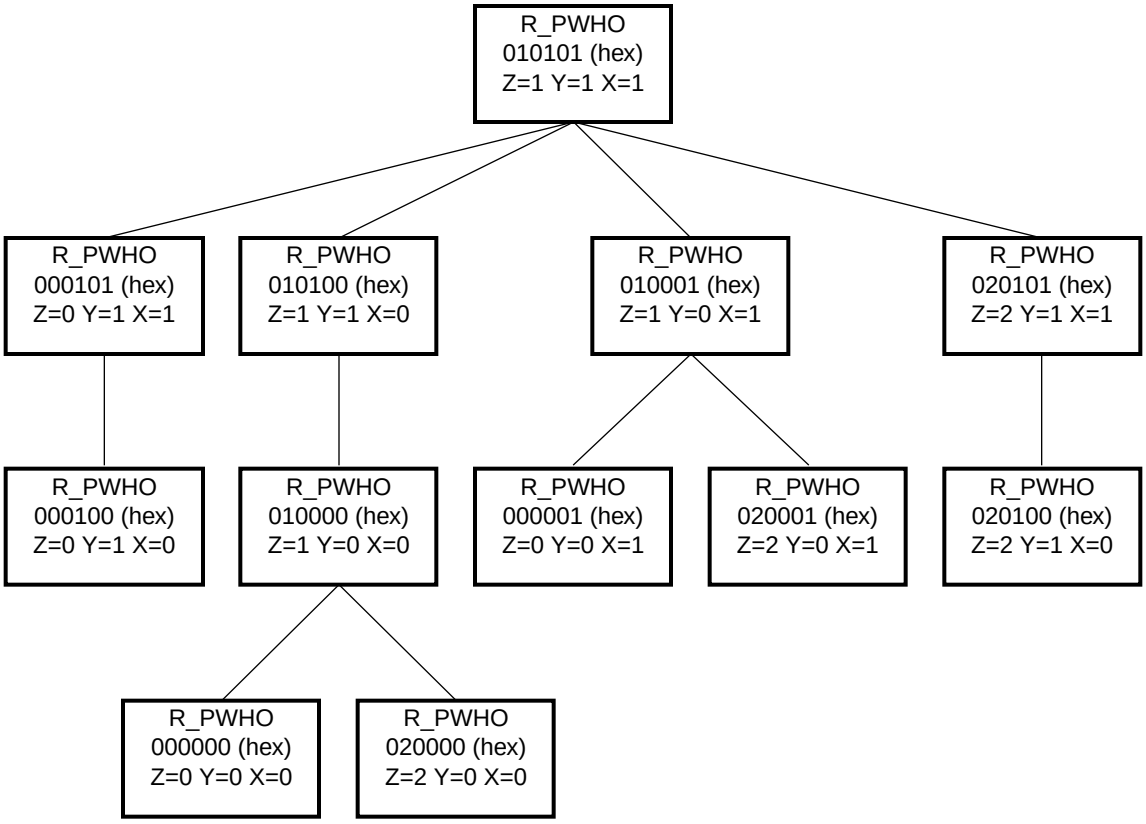
In addition to the root node, software identifies the communication links that transfer B/E control information from each node in the B/E partition to the root node. Figure 22 also shows the communication links that software chose as the B/E communication links. For clarity, Figure 22 does not show the other communication links in B/E partition B of B/E context 3.

Parent-to-child Relationships

After identifying the root node and the communication links that transfer B/E control information from each node in the B/E partition to the root node, software constructs the parent-to-child relationships of the nodes in the B/E partition. The parent-to-child relationships of the nodes indicate how B/E control information transfers from each node in the B/E partition to the root node.

As an example, Figure 23 shows the parent-to-child relationships between the nodes in the B/E partition shown in Figure 22. In this example, node 010100 (hex) is the parent of node 010000 (hex). Node 000001 (hex) is the child of node 010001 (hex).

Figure 23. Parent-to-child Relationships for Figure 22



B/E Partition Configuration

After constructing the parent-to-child relationships of the nodes in the B/E partition, software configures the B/E partition. To do this, software writes the appropriate values in the barrier and eureka configuration (BAR_CFG[31 : 0]) registers of each node in the B/E partition.

Each PE has access to 32 BAR_CFG registers. Each BAR_CFG register controls the partitioning of one B/E context. For example, the BAR_CFG[3] register controls the partitioning of B/E context 3.

Each BAR_CFG register contains three types of bits: a PE bit, child bits, and parent bits (refer to Table 3).

Table 3. BAR_CFG Register Bit Descriptions

Bit Names	Description
PE bit	When set to 1, this bit indicates that this node is part of the B/E partition.
+X child bit	When set to 1, this bit indicates that the node connected to the +X communication link of this node is a child of this node.
–X child bit	When set to 1, this bit indicates that the node connected to the –X communication link of this node is a child of this node.
+Y child bit	When set to 1, this bit indicates that the node connected to the +Y communication link of this node is a child of this node.
–Y child bit	When set to 1, this bit indicates that the node connected to the –Y communication link of this node is a child of this node.
+Z child bit	When set to 1, this bit indicates that the node connected to the +Z communication link of this node is a child of this node.
–Z child bit	When set to 1, this bit indicates that the node connected to the –Z communication link of this node is a child of this node.
Parent bits (4 bits)	These bits indicate which node is the parent of this node or indicate that this node is the root node. 1101 (–3) = The node connected to –Z is the parent of this node. 1110 (–2) = The node connected to –Y is the parent of this node. 1111 (–1) = The node connected to –X is the parent of this node. 0000 (0) = This node is the root node. 0001 (1) = The node connected to +X is the parent of this node. 0010 (2) = The node connected to +Y is the parent of this node. 0011 (3) = The node connected to +Z is the parent of this node.

As an example of BAR_CFG register values, Table 4 shows the BAR_CFG[3] register values for the nodes in the B/E partition shown in Figure 22.

Table 4. BAR_CFG[3] Register Bit Values for the B/E Partition in Figure 22

R_PWHO Value (hex)	Parent Bits	Child Bits						PE Bit
		–Z	+Z	–Y	+Y	–X	+X	
000000 (Z=0 Y=0 X=0)	0011 (+Z)	0	0	0	0	0	0	1
010000 (Z=1 Y=0 X=0)	0010 (+Y)	1	1	0	0	0	0	1
020000 (Z=2 Y=0 X=0)	1101 (–Z)	0	0	0	0	0	0	1
000001(Z=0 Y=0 X=1)	0011 (+Z)	0	0	0	0	0	0	1
010001 (Z=1 Y=0 X=1)	0010 (+Y)	1	1	0	0	0	0	1
020001 (Z=2 Y=0 X=1)	1101 (–Z)	0	0	0	0	0	0	1
000100 (Z=0 Y=1 X=0)	0001 (+X)	0	0	0	0	0	0	1
010100 (Z=1 Y=1 X=0)	0001 (+X)	0	0	1	0	0	0	1
020100 (Z=2 Y=1 X=0)	0001 (+X)	0	0	0	0	0	0	1
000101(Z=0 Y=1 X=1)	0011 (+Z)	0	0	0	0	1	0	1
010101 (Z=1 Y=1 X=1)	0000 (root)	1	1	1	0	1	0	1
020101 (Z=2 Y=1 X=1)	1101 (–Z)	0	0	0	0	1	0	1

Index

P

R

S

W