GBANA DESIGN DOC

Version 0.1.0

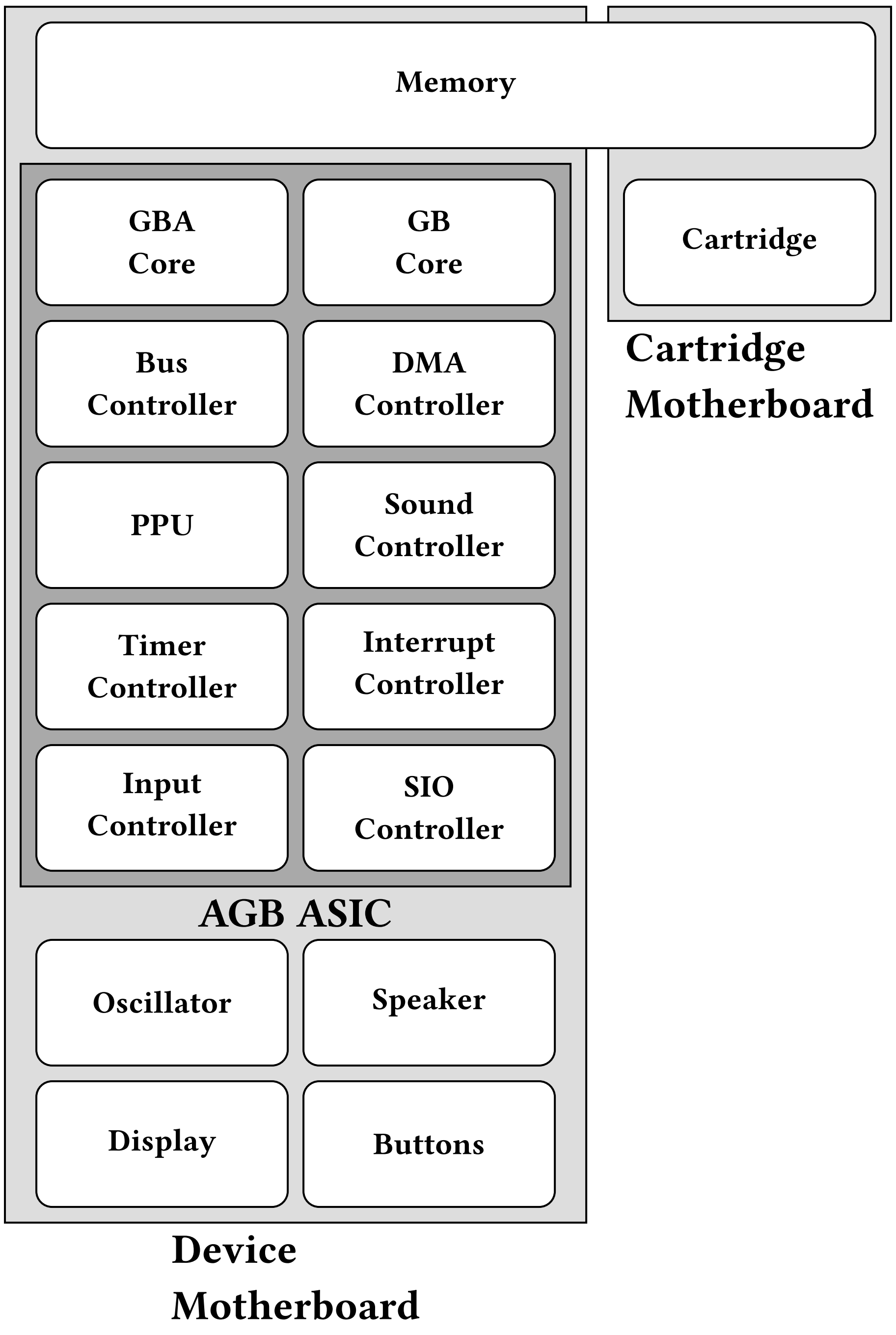
April 25, 2025

Contents

1. Classes

Name	File	Type	Description
GBA Core	`gba_core.odin`	Singleton Class	The ARM7TDMI core inside the AGB ASIC.
GB Core	`gb_core.odin`	Singleton Class	The SM83 core inside the AGB ASIC.
Bus Controller	`bus_controller.odin`	Singleton Class	The bus control circuits inside the AGB ASIC.
DMA Controller	`dma_controller.odin`	Singleton Class	The direct memory access control circuits inside the AGB ASIC.
Sound Controller	`sound_controller.odin`	Singleton Class	The sound control circuits inside the AGB ASIC.
Memory	`memory.odin`	Singleton Class	The internal/device and the external/cartridge memory.
Buttons	`buttons.odin`	Singleton Class	The buttons and adjacent circuits on the device motherboard.
Display	`display.odin`	Singleton Class	The LCD display and adjacent circuits.
GBA Isa	`gba_isa.odin`	Helper Class	Implementation of the ARM4T ISA of the ARM7TDMI core.
GB Isa	`gb_isa.odin`	Helper Class	Implementation of the ISA of the Sharp SM83 core (a hybrid between the 8085 ISA and the Z80 ISA).
PPU	`ppu.odin`	Singleton Class	The Picture Processing Unit / LCD Video Controller inside the AGB ASIC.
Signal	`line_and_bus.odin`	Instance Class	A support class to emulate signals.
SIO Controller	`sio_controller.odin`	Singleton Class	The Serial Input/Output control circuits located inside the AGB ASIC.
Speakers	`speakers.odin`	Singleton Class	The audio output device and adjacent circuits on the device motherboard.
Util	`util.odin`	Helper Class	General utilities.

2. Block Diagram



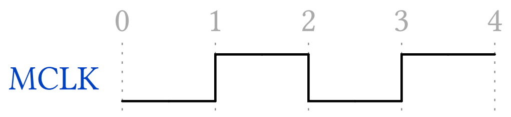
Each rounded reactangle is a **component**. Each component has a state object, which holds its data. Each non-rounded rectangle is a group of components that are semantically related. Components operate concurrently and share data only by means of **signals**.

3. Emulating the Clock & Cycle

Related procedures:

test_main_clock Test procedure.

The core will initially be written as phase-accurate (every phase of every clock cycle is simulated), whose logic will be correct at phase bounds. Then a more high-level core will be implemented, which will be sequence-accurate, whose logic will be correct at sequence bounds, and this core will be verified against the lower-level core.



Main clock frequency: 16 MHz, ie approximately 62.5 ns per cycle. Each phase has two parts: a *start* part, where all the signals are updated and their callback functions are called, and an *interior* part, where the components execute their logic based on their internal state and the updated signals.

4. Signals

Components may only communicate by means of **signals**. There are two ways to affect a signal: (1) by *putting* data on it, and (2) by *forcing* data on it. Forcing updates the output value immediately. Putting schedules an update to the output value, to occur after a certain number of ticks.

Name	Class	Components	Description
`A`	Memory Interface	**Memory** / **Bus Controller**	(Address Bus) The **GBA Core** / **GB Core** writes an address here when it requests memory access.
`ABE`	Bus Controls	**Bus Controller**	(Address Bus Enable) The **GBA Core** sets this to high to become master of the address bus (this is effectively a mutex on `A`), and to low to make the **DMA Controller** master of the address bus.
`ABORT`	Memory Management Interface	**GBA Core**	The **Memory** sets this to high when the requested memory operation cannot be performed, and to low when it can.
`BIGEND`	Bus Controls	**GBA Core**	The **GBA Core** sets this to high to interpret words in memory as being big-endian, and to low to interpret them as being little-endian. On the GBA, this is always 0 (little-endian format).
`BL`	Memory Interface	**Memory** / **Bus Controller**	(Byte Latch) The **GBA Core** writes a bit mask here to indicate which part of the requested word is to be read/written.
`DBE`	Bus Controls	**Bus Controller**	(Data Bus Enable) The **GBA Core** sets this to high to become master of the data output bus (this is effectively a mutex on `DOUT`), and to low to make **DMA Controller** master of the address bus.
`DIN`	Memory Interface	**GBA Core** / **Bus Controller**	(Unidirectional Data Input Bus) The **Memory** writes data here when a read request has been made.
`DOUT`	Memory Interface	**Memory** / **Bus Controller**	(Unidirectional Data Output Bus) The **GBA Core** writes data here when a write request has been made.
`FIQ`	Interrupts	**GBA Core**	(Fast Interrupt Request) Set this to high to request a fast interrupt.
`ISYNC`	Interrupts	**GBA Core**	(Synchronous Interrupt) Set this to high to indicate that the requested interrupt should be synchronous to the processor clock.
`IRQ`	Interrupts	**GBA Core**	(Interrupt Request) Set this to high to request an interrupt.

Name	Class	Components	Description
LOCK	Memory Interface	**Memory** / **Bus Controller**	The **GBA Core** / **GB Core** sets this to high to gain exclusive access to the **Memory** (this is effectively a mutex on the **Memory** signals).
MAS	Memory Interface	**Memory** / **Bus Controller**	(Memory Access Size) The **GBA Core** / **GB Core** writes here the size of the requested data access.
MCLK	Clocks and Timing	**GBA Core**, **Memory**	(Main Clock) The main clock.
M	Processor Mode	**GBA Core**	(Processor Mode) The current mode of the ARM7TDMI core.
MREQ	Memory Interface	**Memory** / **Bus Controller**	(Memory Request) The **GBA Core** / **GB Core** sets this to high to to request memory access in the subsequent cycle.
OPC	Memory Interface	**Memory**	(Op-Code Fetch) The **GBA Core** / **GB Core** sets this to high to indicate that the requested memory access is to fetch the next instruction.
RESET	Bus Controls	**GBA Core**	Set this to high to initialize / restart the AMR7TDMI processor.
RW	Memory Interface	**Memory** / **Bus Controller**	(Read/Write) The **GBA Core** / **GB Core** sets this to high to indicate that the requested memory access is a read, and to low to indicate that it is a write.
SEQ	Memory Interface	**Memory** / **Bus Controller**	(Sequential Cycle) The **GBA Core** / **GB Core** sets this to high to indicate that the subsequent memory cycle will be sequential, and to low to indicate that it will be nonsequential.
TBIT	Processor State	**GBA Core**	(Thumb Mode Bit) Set this to high to switch the ARM7TDMI core to Thumb mode, and to low to switch it to ARM mode.
WAIT	Clocks and Timing	**GBA Core**	Set this to high to insert a wait cycle.

5. Sequences

There are two fundamental types of sequence: **internal sequence** and **external sequence**. Internal cycles are initiated by the **GBA Core** by calling a `gba_initiate_<cycle_name>_cycle_request` procedure and then one or more other components respond by interpreting the signals and calling a `<component_name>_initiate_<cycle_name>_cycle_response` procedure.

Types of intervals in a timing diagram:

- *Open Unshaded* - The line/bus is expected to remain stable throughout this interval.
  - *Writing* occurs at the *start* and is prohibited in the *interior*.
  - *Reading* is allowed at the *start* (by signals succeding it in the tick order) and in the *interior*.
- *Open Shaded* - The line/bus is expected to change at an arbitrary time during this interval.
  - *Writing* is prohibited at the *start* and allowed in the *interior*.
  - Reading is allowed at the *start* and prohibited in the *interior*.
- *Closed* - The line/bus is disabled.
  - *Writing* is prohibited at the *start* and prohibited in the *interior*.
  - *Reading* is prohibited at the *start* and in the *interior*.

In request/response contexts, request data is in displayed in blue, and respone data is displayed in pink.

5.1. Memory Sequence

Related Procedures
test_memory_sequence
gba_request_memory_sequence
memory_respond_memory_sequence

Request Signals	Response Signals
MREQ , SEQ , RW , A , DOUT	WAIT , ABORT , DIN

The Memory Sequence is an external sequence where the GBA Core requests and the Memory responds. Reads and writes have distinct logic. Word-wide bus access, halfword-wide bus access, and byte-wide bus access have distinct logic. The Memory may extend the time to fulfill the request and it may assert that a request may not be fulfilled.

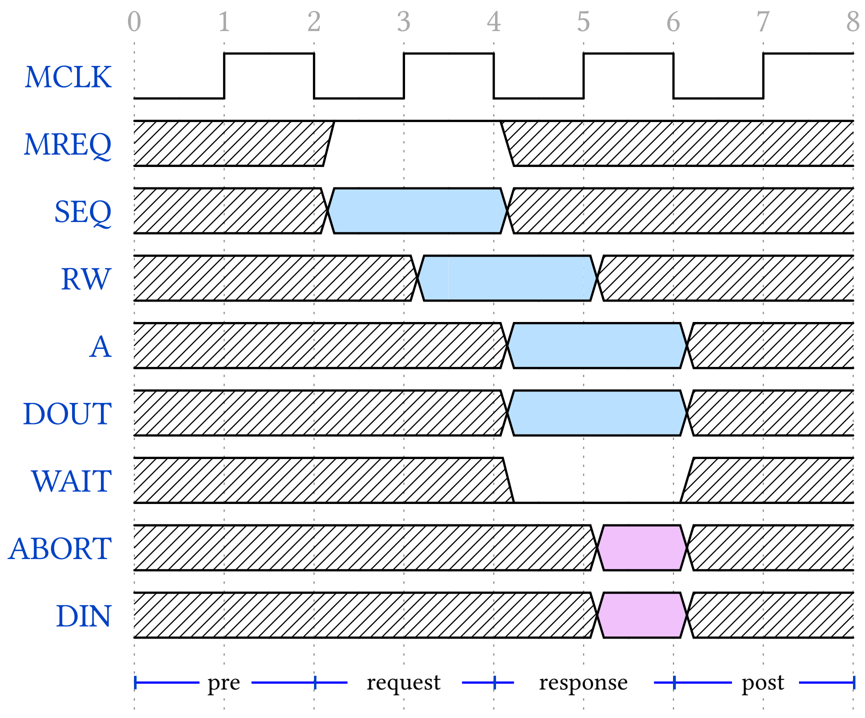


Figure 1: Memory Sequence timing diagram

- The **GBA Core** must set SEQ to high when the access is sequential to the access performed in the previous cycle.
- The **GBA Core** must set RW to high for reading and to low for writing.
- The **GBA Core** must write the address to A .
- The **Memory** may set WAIT to high to delay the response cycle.
- The **Memory** may set ABORT to high to indicate that the request cannot be fulfilled.
- The **Memory** must put the data on DIN during the high phase of the response cycle.

5.2. Nonsequential Memory Sequence (N-Cycle)

Related Procedures
`test_N_cycle`
`gba_request_N_cycle`
`memory_respond_N_cycle`

Request Signals	Response Signals
`MREQ`, `SEQ`, `RW`, `A`, `DOUT`	`WAIT`, `ABORT`, `DIN`

Duration: 1 to 2 clock cycles

The Nonsequential Memory Sequence (or N-Cycle) is a memory access sequence, preceded by an internal sequence or another memory access sequence to an address other than the address immediately before the current address.

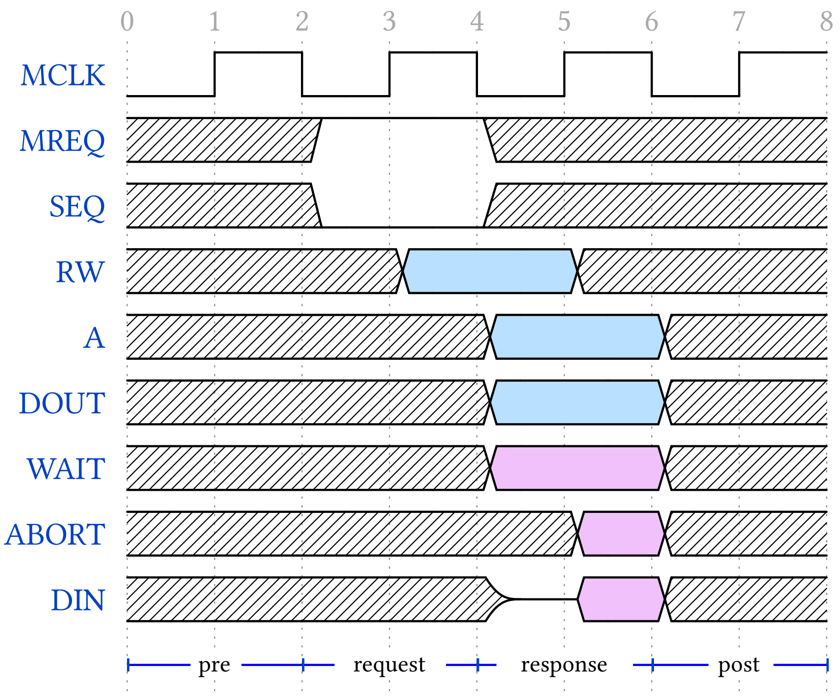


Figure 2: Nonsequential Memory Sequence timing diagram

- The **GBA Core** must set `RW` to high for reading and to low for writing.
- The **GBA Core** must write the address to `A`.
- The **Memory** may set `WAIT` to high to delay the response cycle.
- The **Memory** may set `ABORT` to high to indicate that the request cannot be fulfilled.
- The **Memory** must put the data on `DIN` during the high phase of the response cycle.

5.3. Sequential Memory Sequence (S-Cycle)

Related Procedures
test_S_cycle
gba_request_S_cycle
memory_respond_S_cycle

Request Signals	Response Signals
MREQ , SEQ , RW , A , DOUT	WAIT , ABORT , DIN

Duration: 2 clock cycles

The Sequential Memory Sequence (or S-Cycle) is a memory access sequence, preceded by another memory access sequence to the address immediately before the current address.

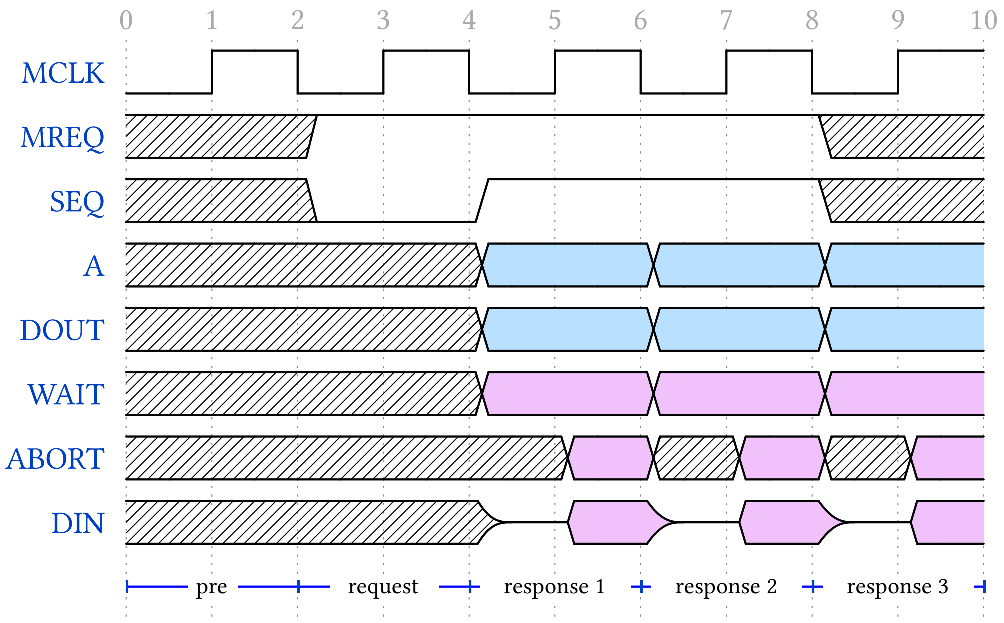


Figure 3: Sequential Memory Sequence timing diagram

- The **GBA Core** must set RW to high for reading and to low for writing.
- The **GBA Core** must write the address to A .
- The **Memory** may set WAIT to high to delay the next response cycle.
- The **Memory** may set ABORT to high to indicate that the request cannot be fulfilled.
- The **Memory** must put the data on DIN during the high phase of the response cycle.

5.4. Internal Sequence

Related Procedures
test_I_cycle
gba_initiate_I_cycle

Signals
MREQ , SEQ , A , DIN , DOUT

Duration: 1 clock cycle

The Internal Sequence (or I-Cycle) is a sequence that doesn't involve exchaning data with any components outside of the core.

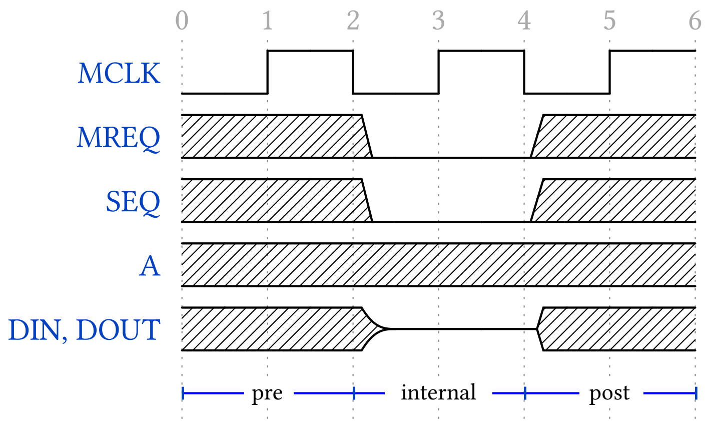


Figure 4: Internal Sequence timing diagram

5.5. Merged Internal-Sequential Sequence (MIS-Cycle)

Related Procedures
test_MIS_cycle
gba_request_MIS_cycle
memory_respond_MIS_cycle

Request Signals	Response Signals
MREQ , SEQ , RW , A , DOUT	WAIT , ABORT , DIN

Duration: 2 clock cycles

The Merged Internal-Sequential Sequence (or Merged IS-Cycle) is an internal sequence followed immediately by a sequential memory cycle.

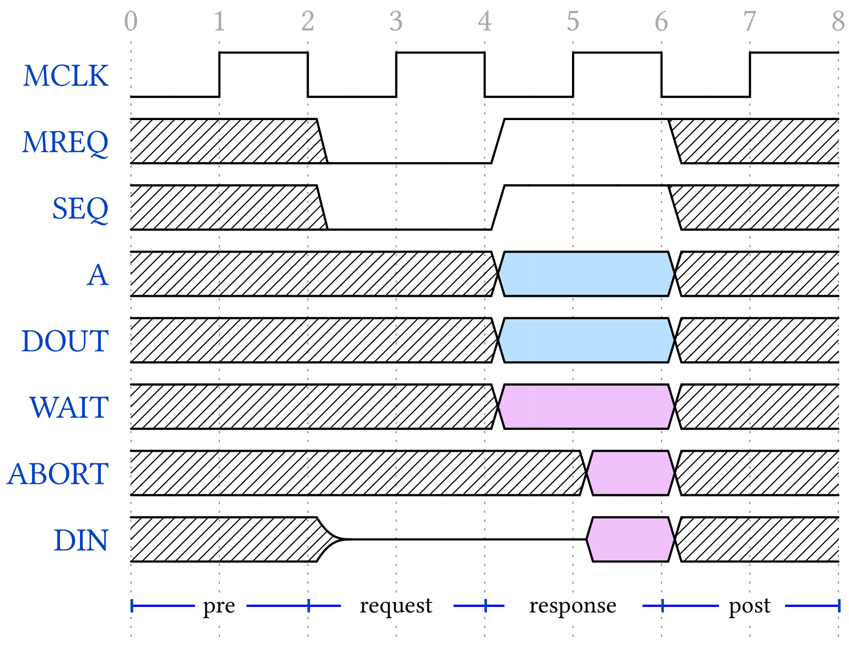


Figure 5: Merged Internal-Sequential Sequence timing diagram

5.6. Depipelined Addressing

Related Procedures
test_depipelined_addressing

Request Signals	Response Signals
MREQ , SEQ , RW , A , DOUT	WAIT , ABORT , DIN

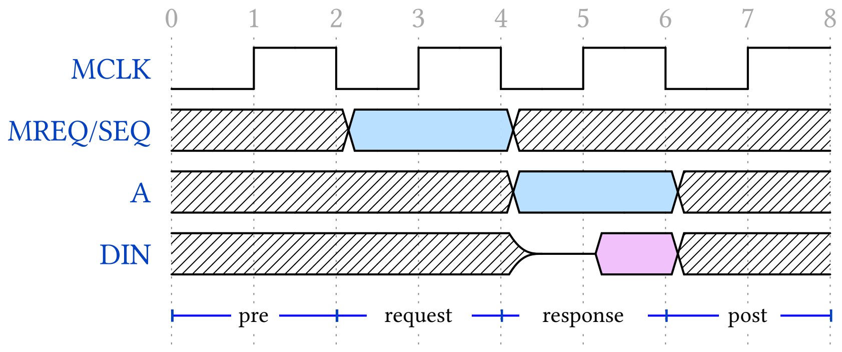


Figure 6: Depipelined Addressing timing diagram

5.7. Data Write Sequence (DW-Cycle)

Related Procedures
`test_DW_cycle`
`gba_request_DW_cycle`
`memory_respond_DW_cycle`

Request Signals	**Response Signals**
`MREQ` , `RW` , `A` , `DOUT`	`WAIT` , `ABORT`

Duration: 2 clock cycles

A Data Write Sequence is an external sequence where a write operation is requested by the GBA Core.

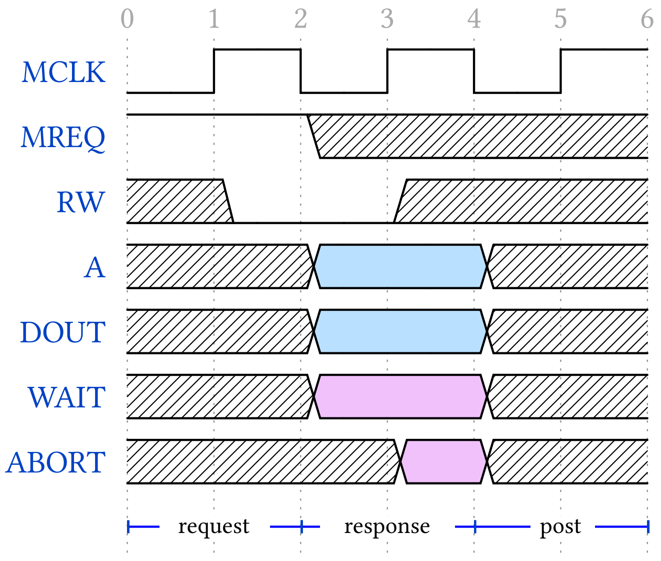


Figure 7: Data Write Sequence timing diagram

5.8. Data Read Sequence (DR-Cycle)

Related Procedures
test_DR_cycle
gba_request_DR_cycle
memory_respond_DR_cycle

Request Signals	Response Signals
MREQ , RW , A , DOUT	WAIT , ABORT

Duration: 2 clock cycles

A Data Read Sequence is an external sequence where a read operation is requested by the GBA Core.

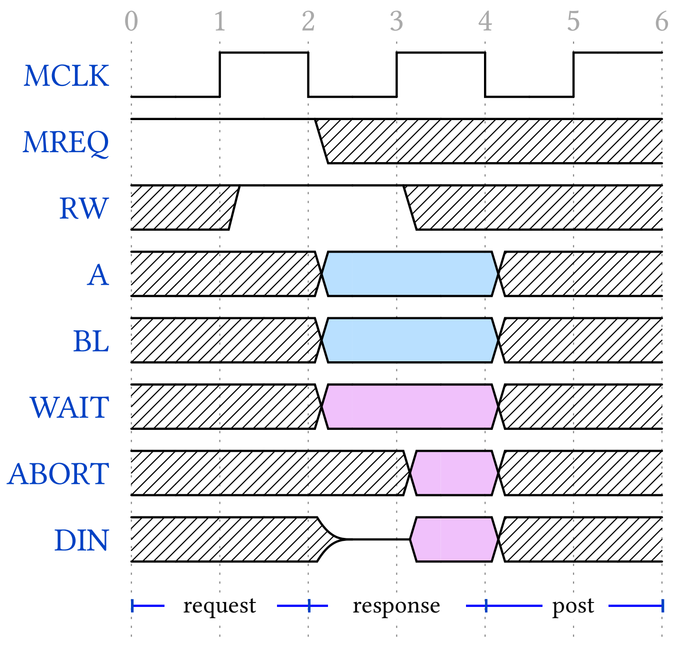


Figure 8: Data Read Sequence timing diagram

5.9. Halfword-Wide Memory Sequence

Related Procedures
`test_halfword_memory_sequence`
`gba_request_halfword_memory_sequence`
`memory_respond_halfword_memory_sequence`

Request Signals	Response Signals
`MREQ`, `SEQ`, `RW`, `A`, `DOUT`	`WAIT`, `ABORT`, `DIN`

The Halfword-Wide Memory Sequence is the same as the basic Memory Sequence, except for memory with a 16-bit wide bus, thus requiring 2 cycles per word.

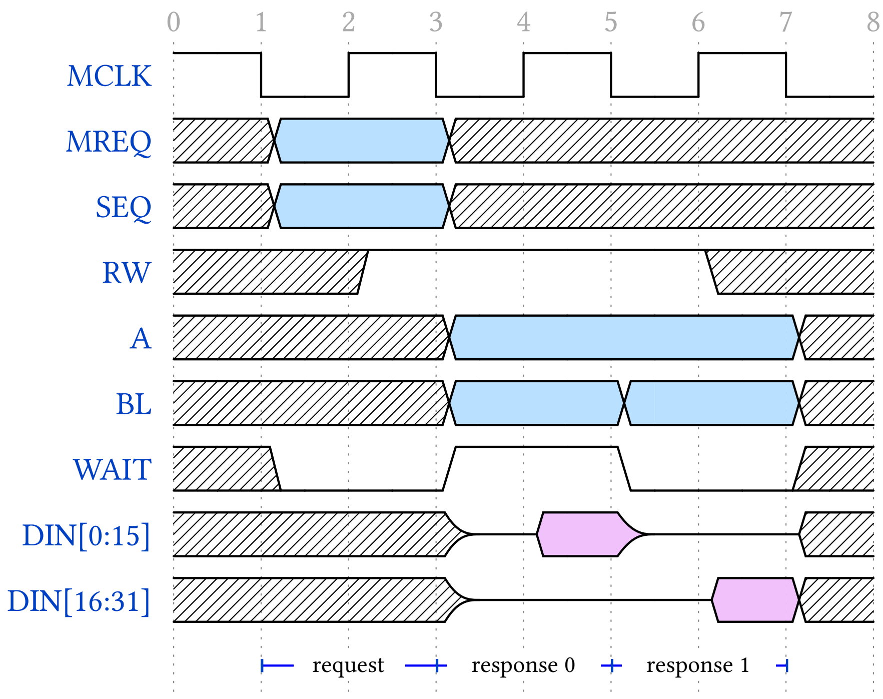


Figure 9: Halfword-Wide Read Memory Sequence timing diagram

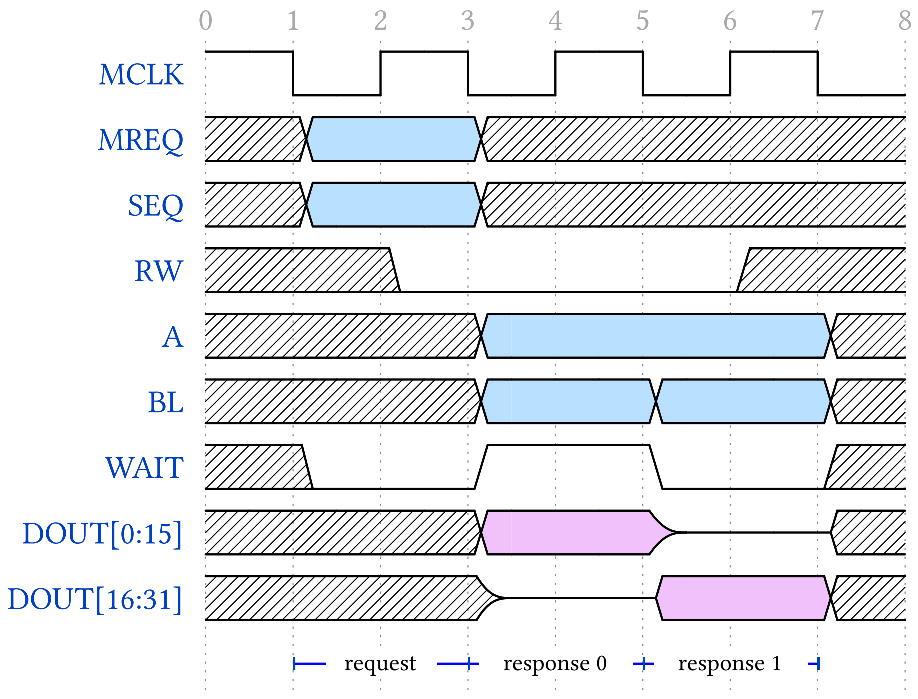


Figure 10: Halfword-Wide Write Memory Sequence timing diagram

5.10. Byte-Wide Memory Sequence

Related Procedures
test_byte_memory_sequence
gba_request_byte_memory_sequence
memory_respond_byte_memory_sequence

Request Signals	Response Signals
MREQ , SEQ , RW , A , DOUT	WAIT , ABORT , DIN

The Byte-Wide Memory Sequence is the same as the basic Memory Sequence, except for memory with an 8-bit wide bus, thus requiring 4 cycles per word.

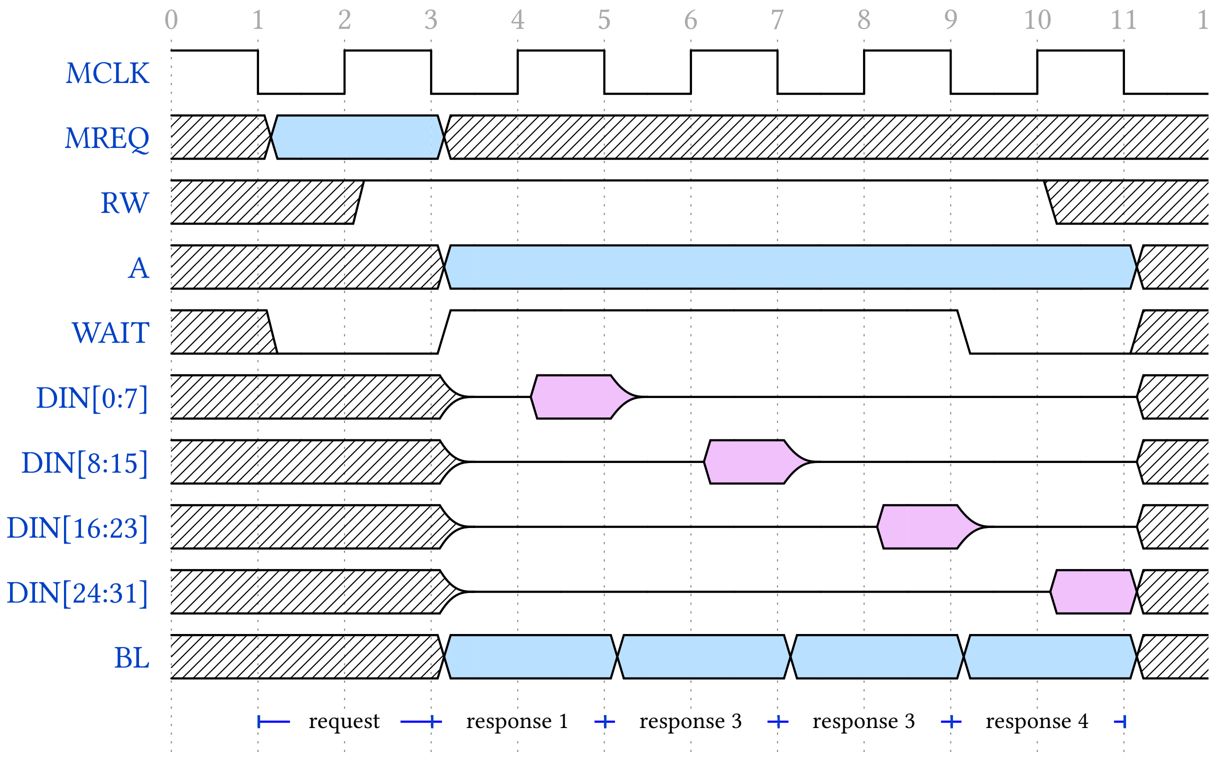


Figure 11: Byte-Wide Read Memory Sequence timing diagram

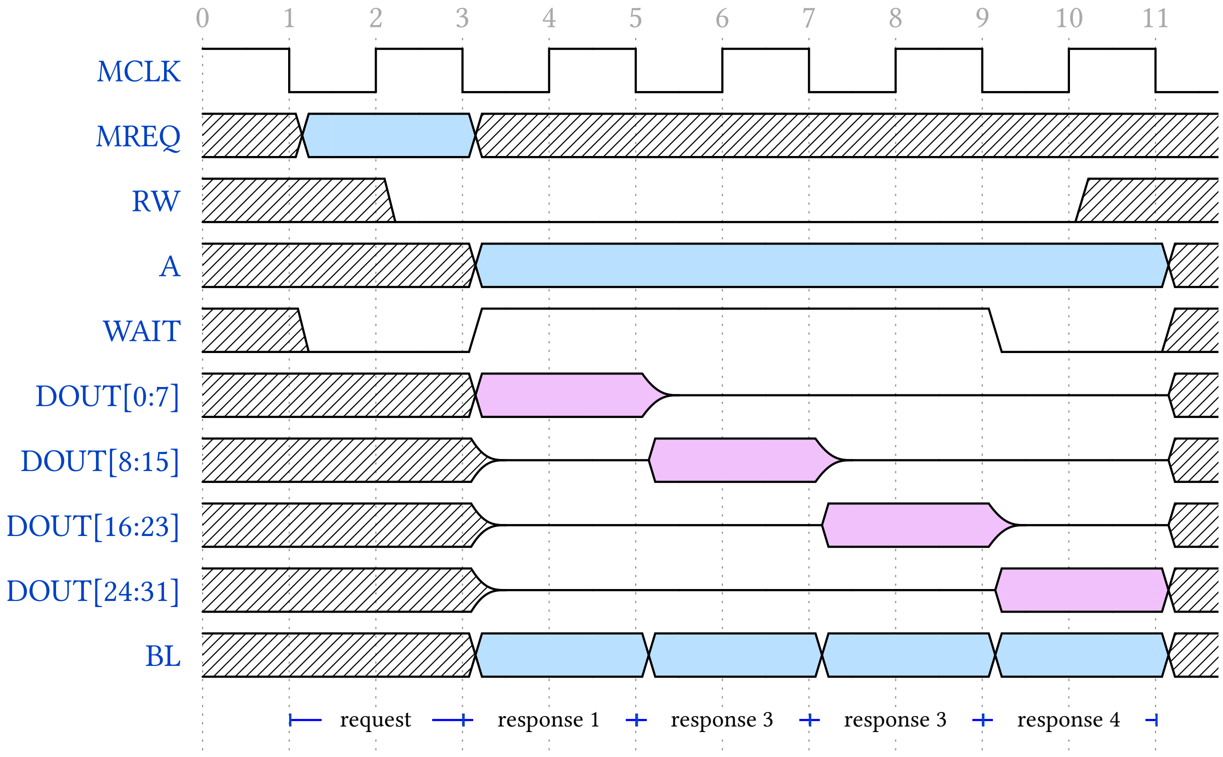


Figure 12: Byte-Wide Write Memory Sequence timing diagram

5.11. Reset Sequence (RS-Cycle)

Related Procedures
test_RS_cycle
gba_initiate_RS_cycle

Request Signals	Response Signals
MREQ , SEQ , RW , EXEC , OPC , A	WAIT , ABORT , DIN

Duration: 5 clock cycles

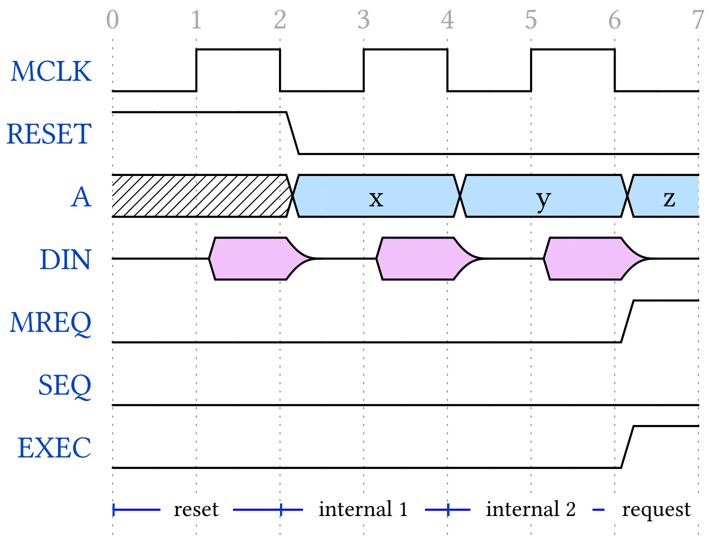


Figure 13: Reset Sequence timing diagram

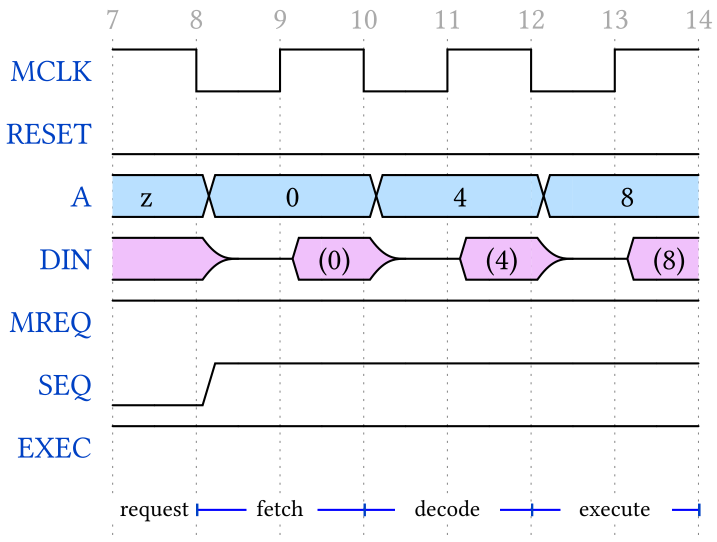


Figure 14: Reset Sequence timing diagram (continued)

5.12. General Timing

Related Procedures
test_general_timing

Phase 1 Signals	Phase 2 Signals
MREQ , SEQ , EXEC , EXEC , INSTRVALID , A , BIGEND , ISYNC	RW , MAS , LOCK , M , TBIT , OPC , ISYNC

The General Timing diagram defines in which phase of the cycle each of the control signals is allowed to change.

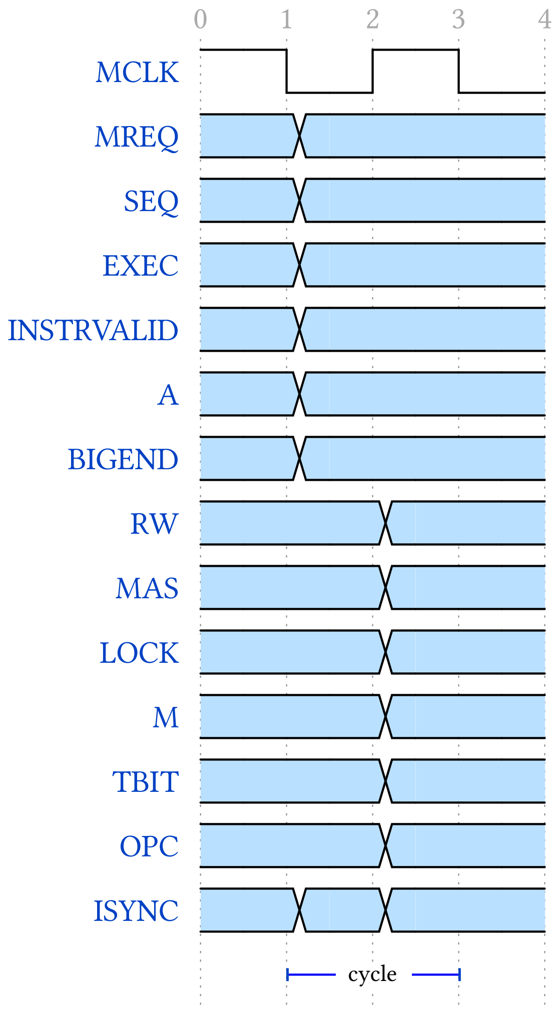


Figure 15: General timing diagram

5.13. Address Bus Control

Related Procedures
test_address_bus_control

Signals
ABE , A , RW , LOCK , OPC , MAS

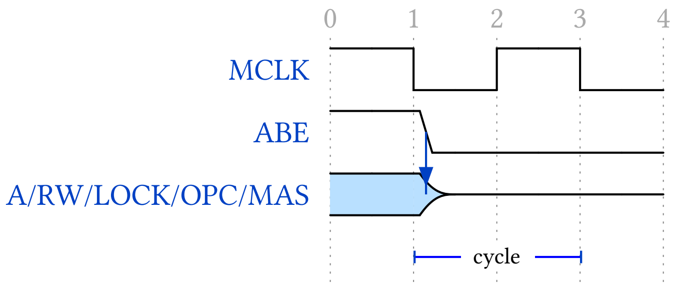


Figure 16: Address Bus Control timing diagram

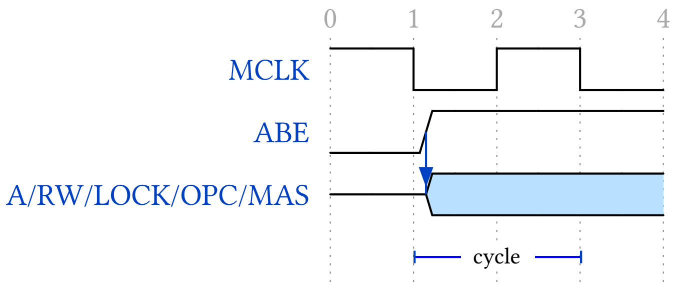


Figure 17: Address Bus Control timing diagram

- ABE may change during phase 1.
- A , RW , LOCK , OPC , and MAS are enabled/disabled immediately when ABE switches to high/low.
- A , RW , LOCK , OPC , and MAS must be stable at the starts of both phases.

5.14. Data Bus Control

Related Procedures
test_data_bus_control

Signals
DBE , DIN , DOUT

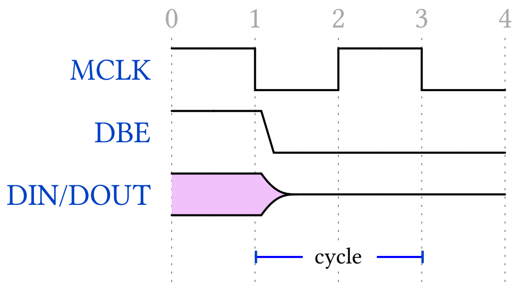


Figure 18: Data Bus Control timing diagram

5.15. Exception Control

Related Procedures
test_exception_control

Signals
ABORT , FIQ , IRQ , RESET

The exceptional behaviors are: (1) aborted memory access, (2) interrupt, (3) fast interrupt, (4) reset.

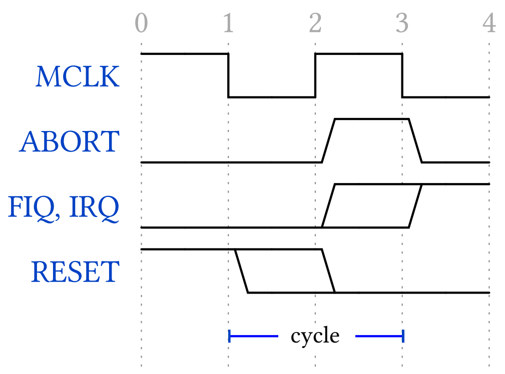


Figure 19: Exception Control timing diagram

1. FIQ and IRQ signals must be set one cycle ahead of the cycle in which they'll be handled, during the high phase, and they must remain stable through the start of the low phase of the next cycle.

5.16. Address Pipeline Control

Related Procedures
test_address_pipeline_control

Signals
APE , A , RW , LOCK , OPC , MAS

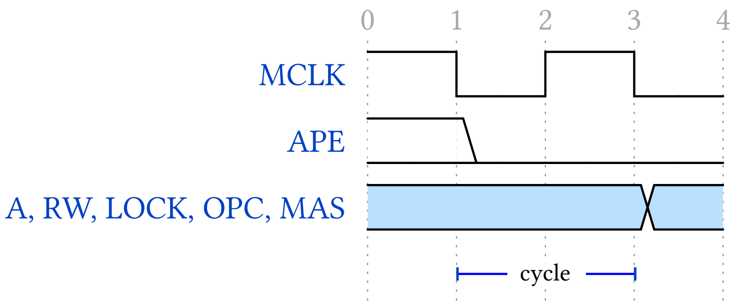


Figure 20: Address Pipeline Control timing diagram

5.17. General Instruction Cycle

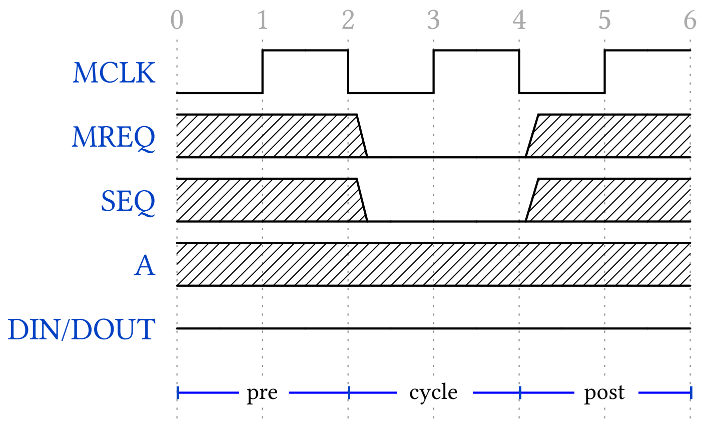


Figure 21: General Instruction Cycle timing diagram

1. There are two types of request signals: request type signals and request address signals, which are broadcast at least one tick ahead of the response cycle.
2. The request type signals (`MREQ` and `SEQ`) are pipelined up to 2 ticks ahead of the cycle to which they apply.
3. The request address signals (`A` , `MAS` , `RW` , `OPC` , and `TBIT`) are pipelined up to 1 tick ahead of the cycle to which they apply.
4. The instruction cycle is the response cycle.
5. When `OPC` is high, the address is incremented each cycle (epistemic status: *guess*).

5.18. Branch and Branch with Link Instruction Cycle (BABLI-Cycle)

Related Instructions
`B`, `BL`

Related Procedures
`test_BABLI_cycle`
`gba_request_BABLI_cycle`
`memory_respond_BABLI_cycle`

Duration: 3 clock cycles

	Parameters
`pc`	Program counter, before executing the instruction.
`L`	Instruction length, 4 for ARM state, 2 for Thumb state.
`alu`	The instruction operand—ie, the address to jump to.
`i`	MAS, 2 for ARM state, 1 for Thumb state.

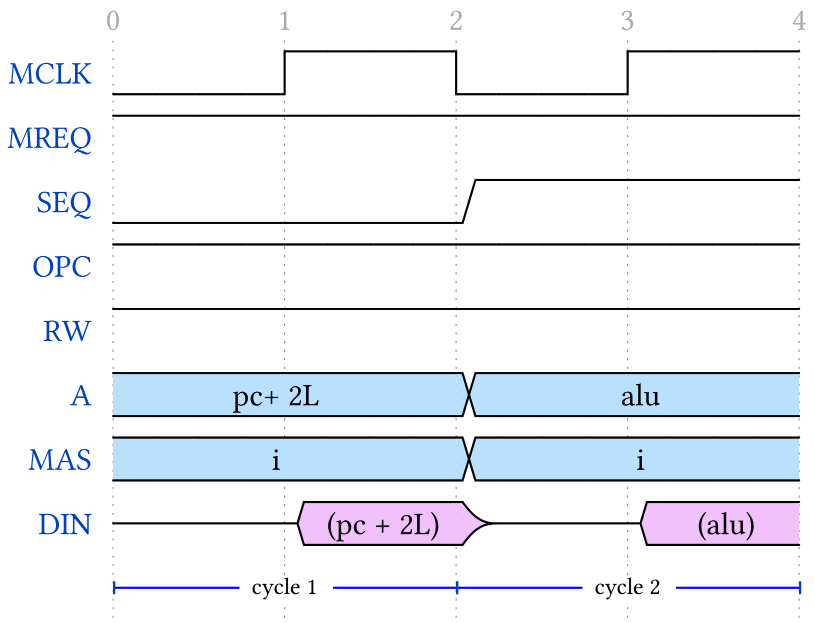


Figure 22: Branch and Branch with Link Instruction Cycle

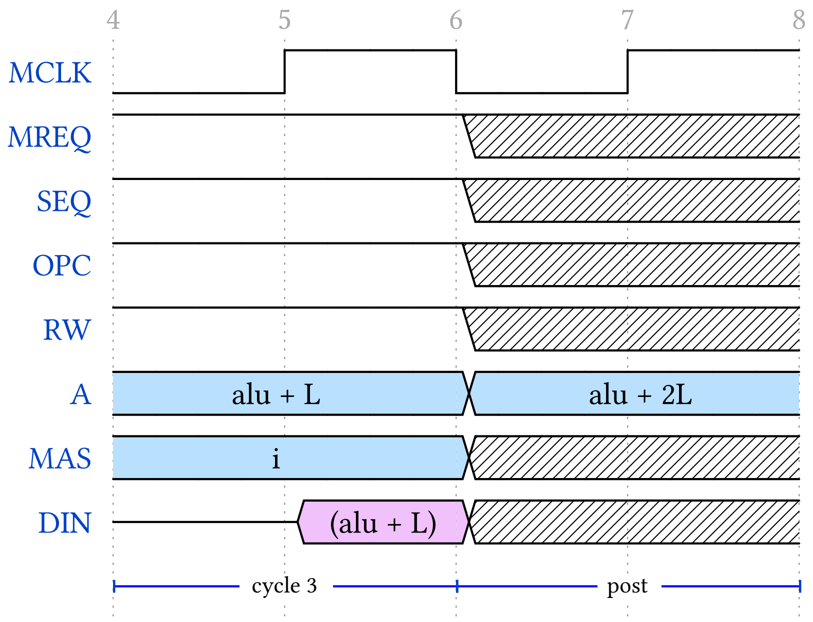


Figure 23: Branch and Branch with Link Instruction Cycle (continued)

5.19. Thumb Branch with Link Instruction Cycle (TBLI-Cycle)

Related Instructions

Related Procedures
test_TBLI_cycle
gba_request_TBLI_cycle
memory_respond_TBLI_cycle

Duration: 4 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
alu	The instruction operand—ie, the address to jump to.

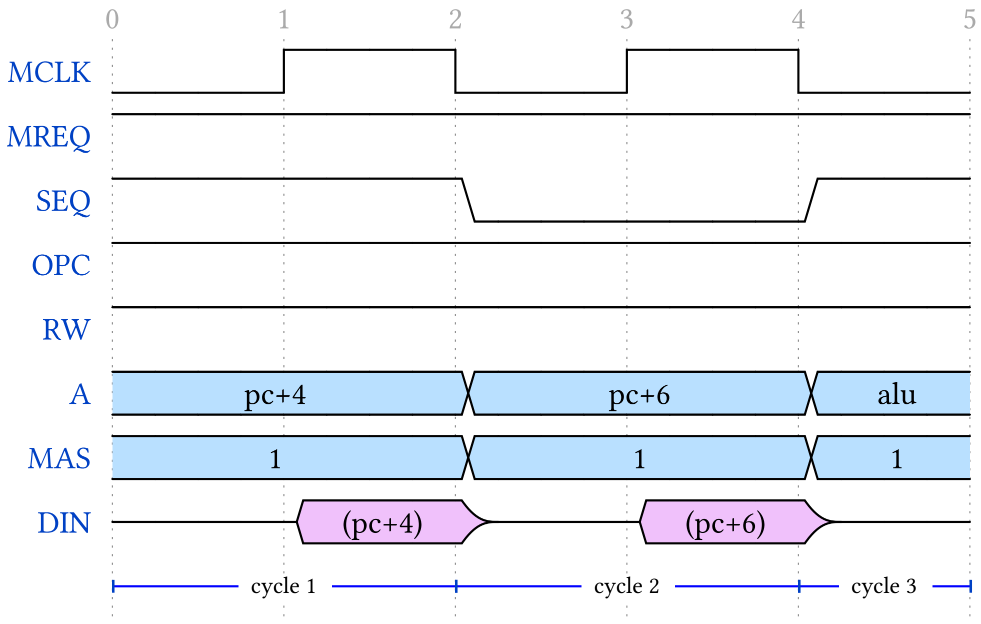


Figure 24: Thumb Branch with Link Instruction Cycle

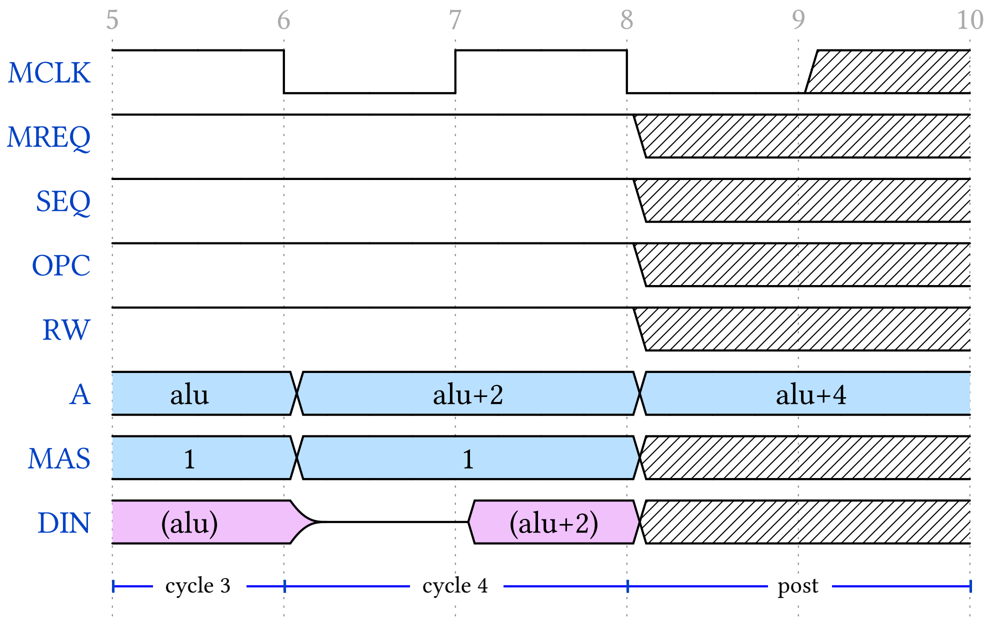


Figure 25: Thumb Branch with Link Instruction Cycle (continued)

5.20. Branch and Exchange Instruction Cycle (BAEI-Cycle)

Related Instructions
BX

Related Procedures
test_BAEI_cycle
gba_request_BAEI_cycle
memory_respond_BAEI_cycle

Duration: 3 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
alu	The instruction operand—ie, the address to jump to.
I	MAS before executing the instruction.
i	MAS after executing the instruction.
W	Instruction width before executing the instruction.
w	Instruction width after executing the instruction.
T	TBIT before executing the instruction.
t	TBIT after executing the instruction.

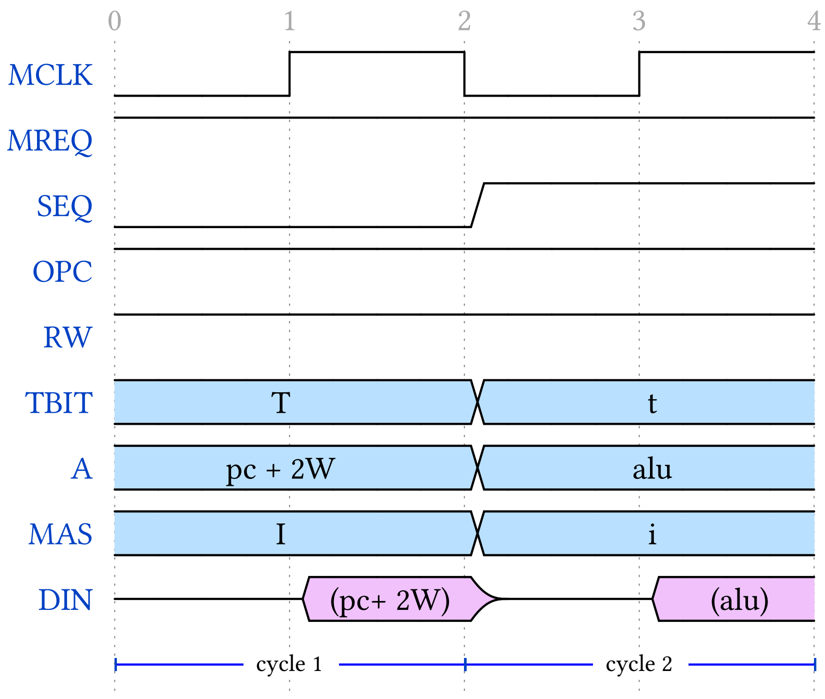


Figure 26: Branch and Exchange Instruction Cycle

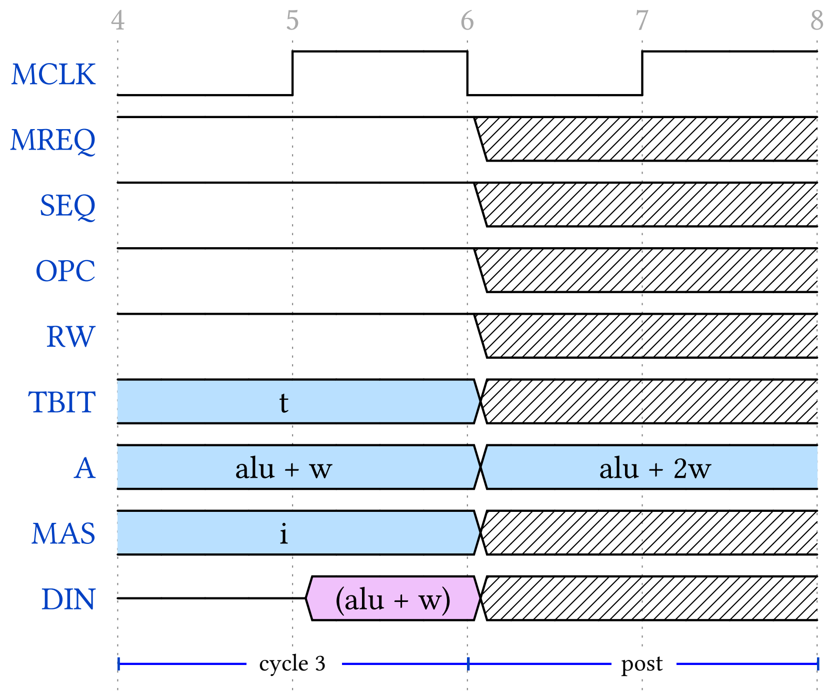


Figure 27: Branch and Exchange Instruction Cycle (continued)

5.21. Data Processing Instruction Cycle (DPI-Cycle)

Related Instructions
ADC , ADD , AND , BIC , CMN , CMP , EOR , MOV , MRS , MSR , MVN , ORR , RSB , RSC , SBC , SUB , TEQ , TST

Related Procedures
test_DPI_cycle
gba_request_DPI_cycle
memory_respond_DPI_cycle

Duration: 1 to 4 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
alu	The instruction operand—ie, the address of the shifter operand.
i	MAS, 2 for ARM state, 1 for Thumb state.

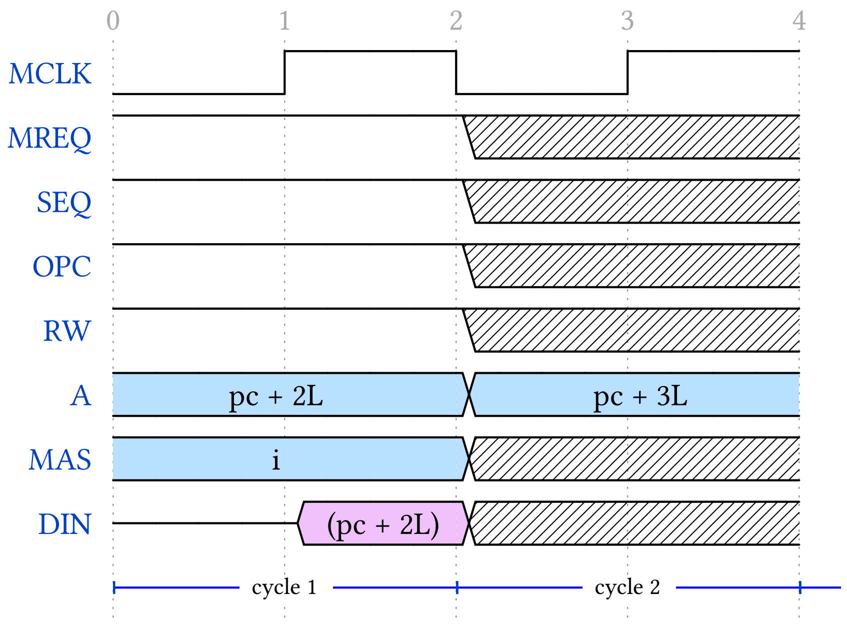


Figure 28: Data Processing Instruction Cycle (normal)

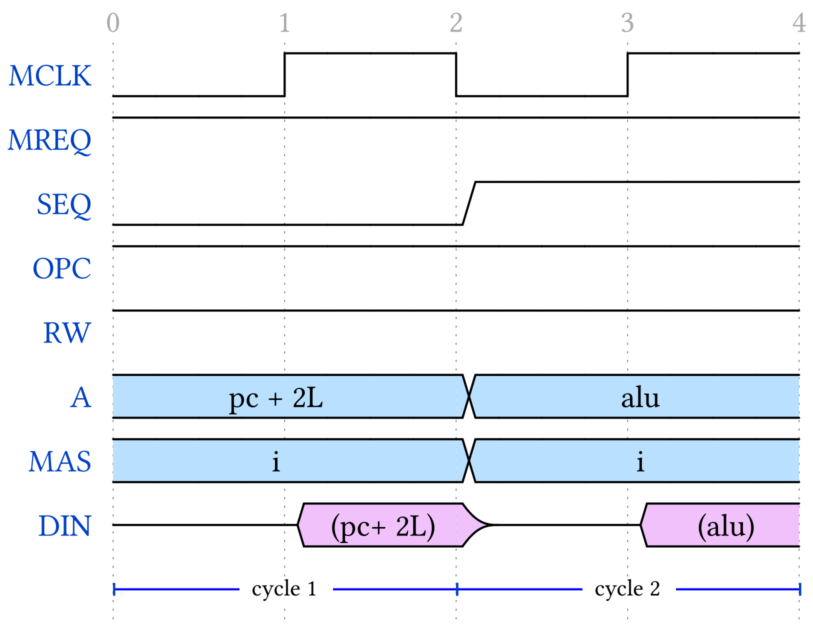


Figure 29: Data Processing Instruction Cycle (dest=pc)

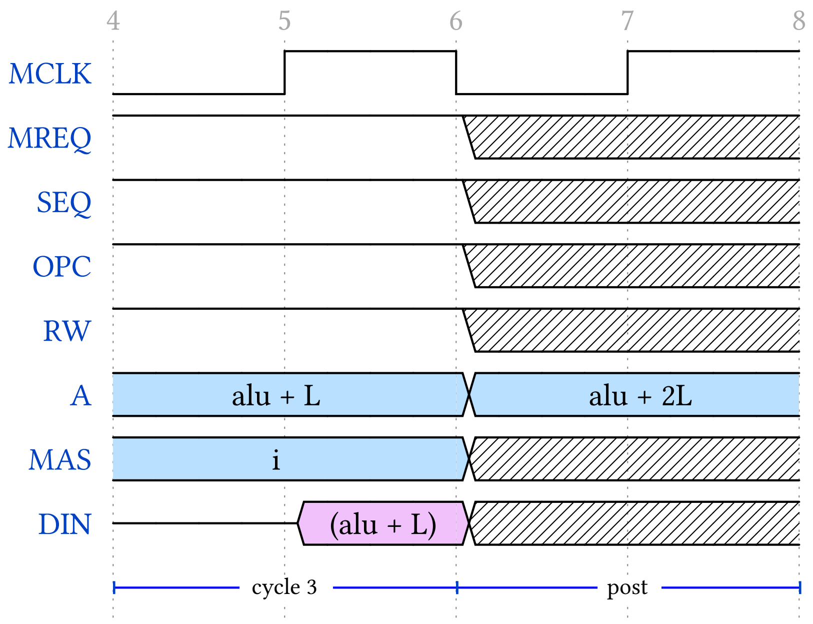


Figure 30: Data Processing Instruction Cycle (dest=pc) (continued)

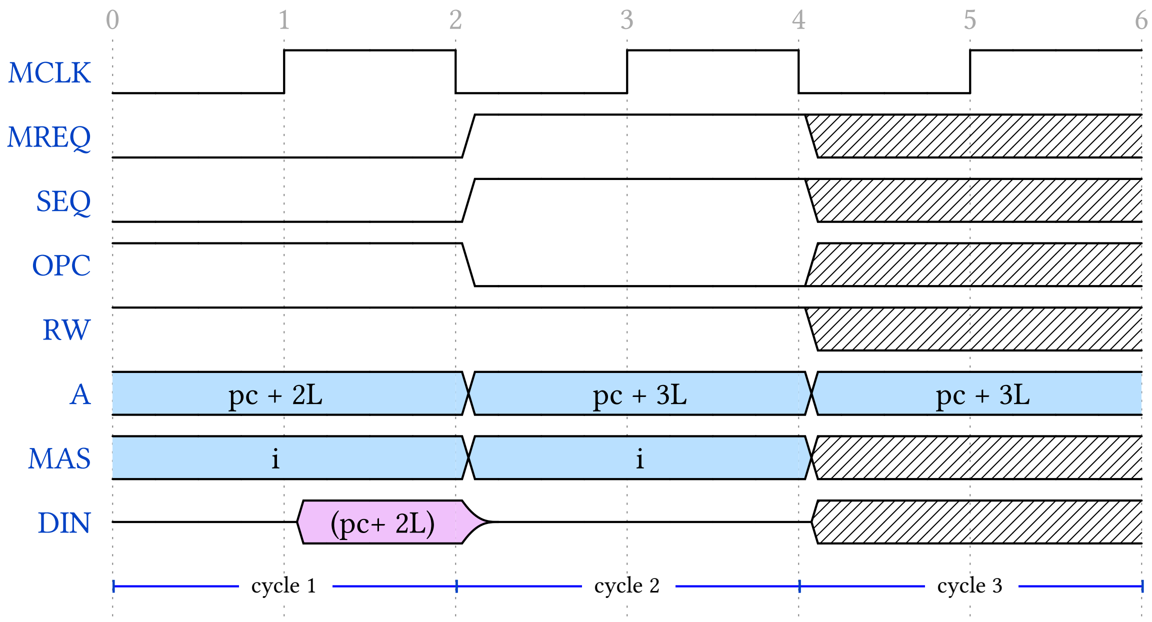


Figure 31: Data Processing Instruction Cycle (shift(RS))

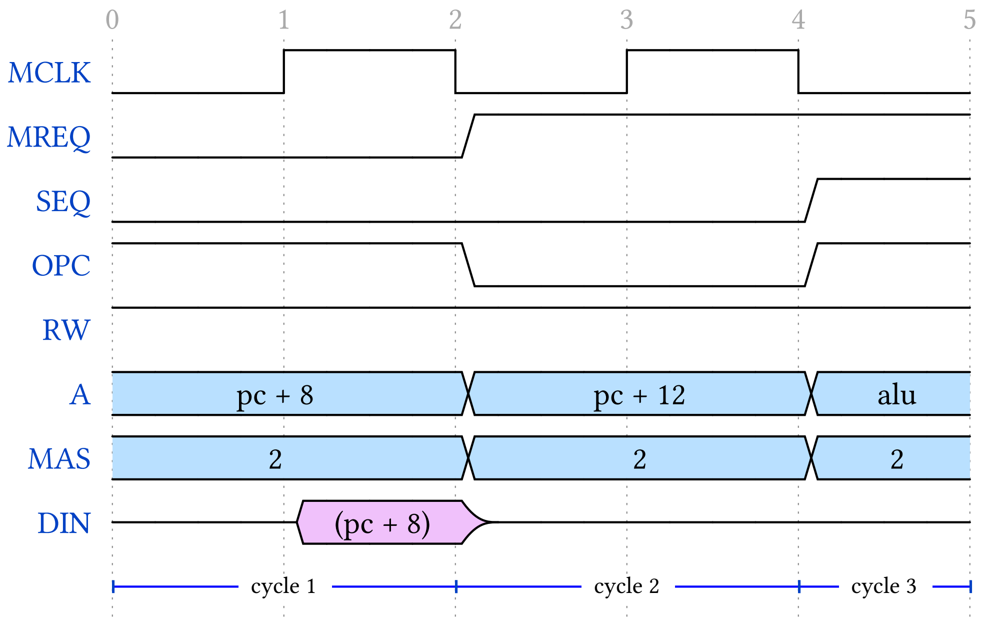


Figure 32: Data Processing Instruction Cycle (shift(Rs) dest=pc)

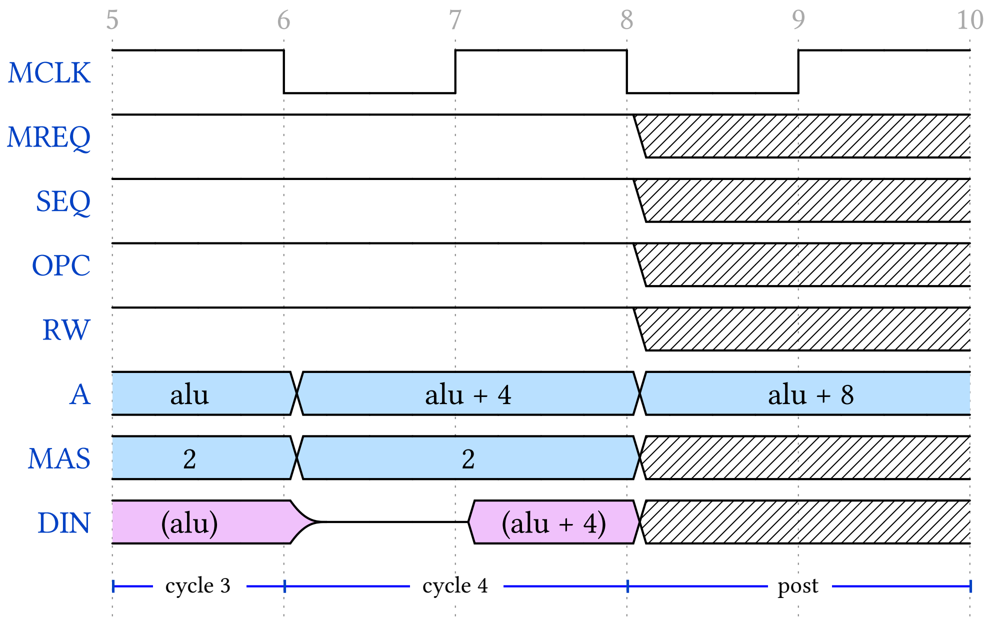


Figure 33: Data Processing Instruction Cycle (shift(Rs) dest=pc) (continued)

5.22. Multiply and Multiply Accumulate Instruction Cycle (MAMAI-Cycle)

Related Instructions
MLA , MUL , SMLAL , SMULL , UMLAL , UMULL

Related Procedures
test_MAMAI_cycle
gba_request_MAMAI_cycle
memory_respond_MAMAI_cycle

Duration: 3 to 4 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
i	MAS, 2 for ARM state, 1 for Thumb state.

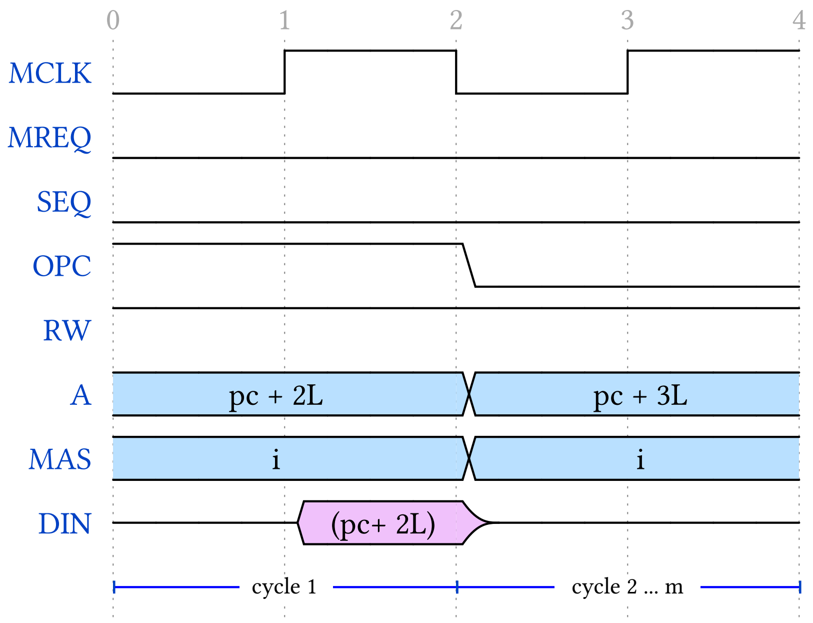


Figure 34: Multiply Instruction Cycle

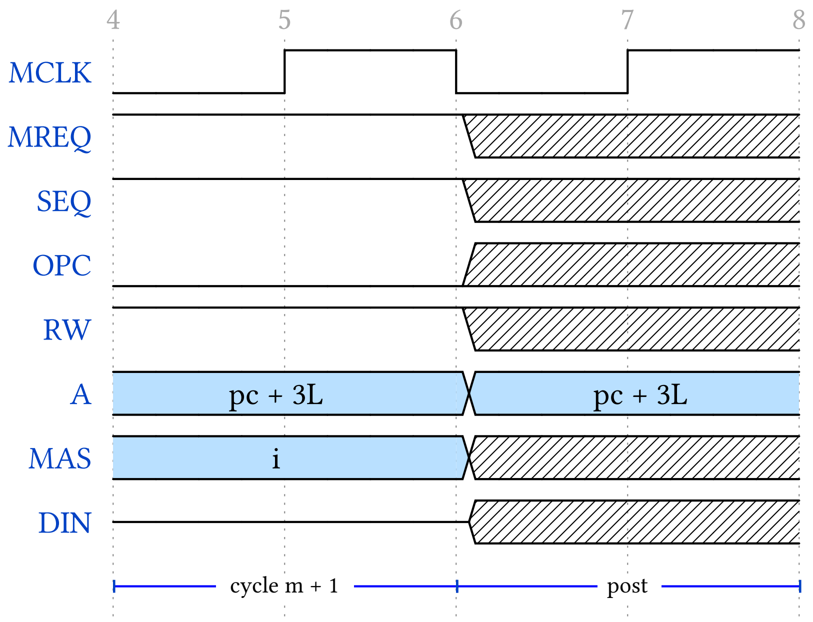


Figure 35: Multiply Instruction Cycle (continued)

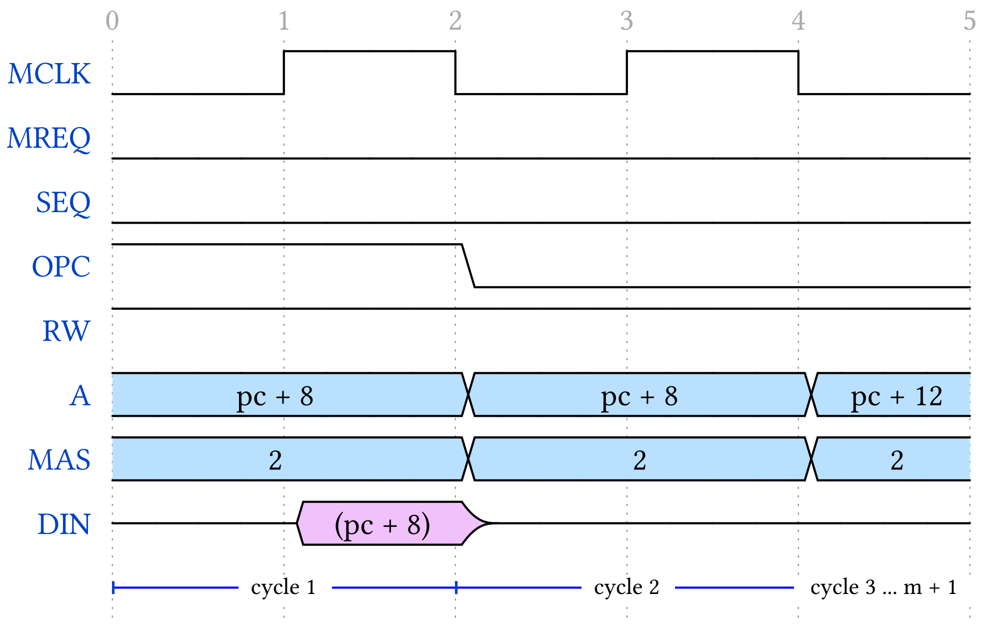


Figure 36: Multiply Accumulate Instruction Cycle

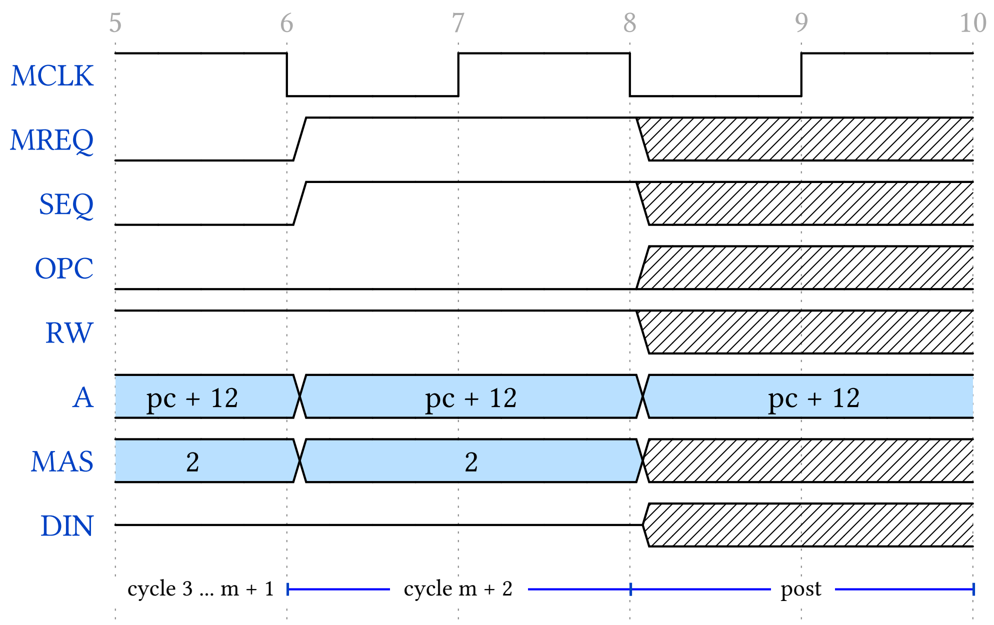


Figure 37: Multiply Accumulate Instruction Cycle (continued)

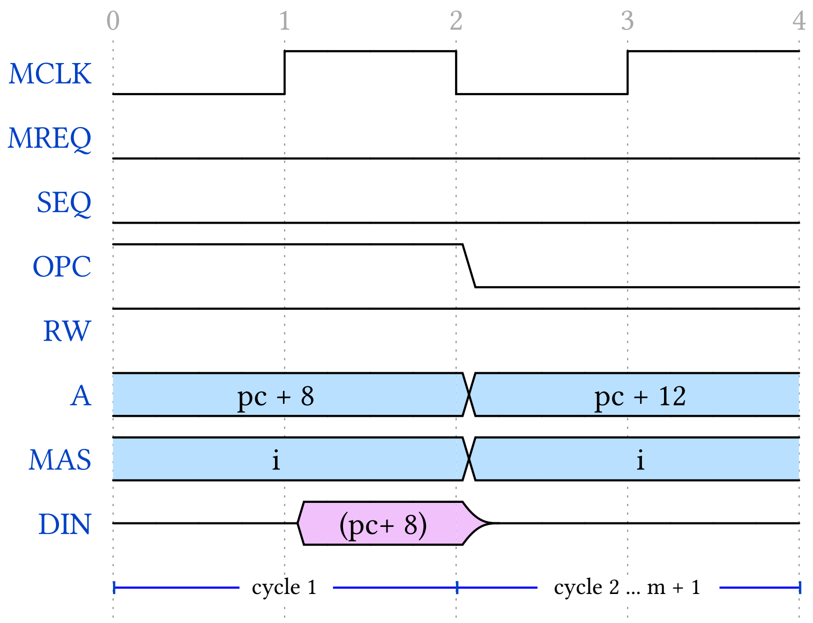


Figure 38: Multiply Long Instruction Cycle

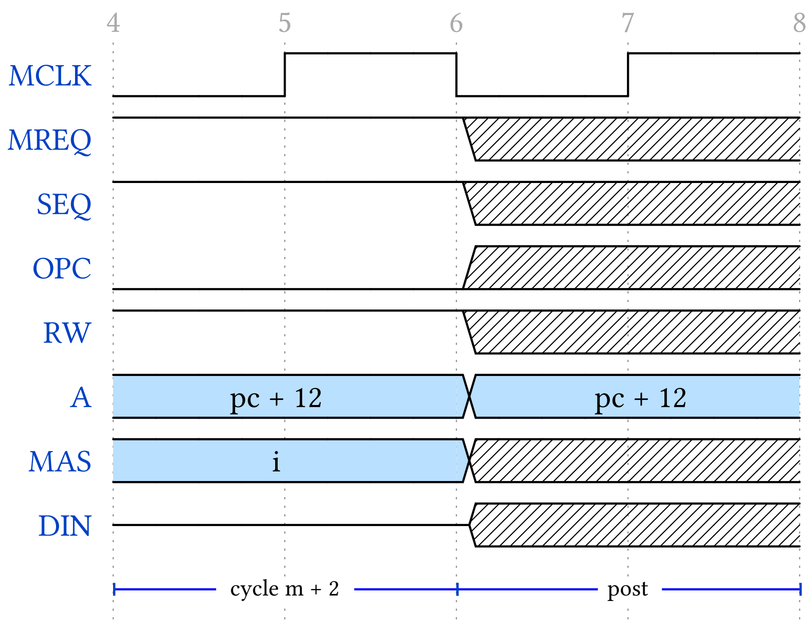


Figure 39: Multiply Long Instruction Cycle (continued)

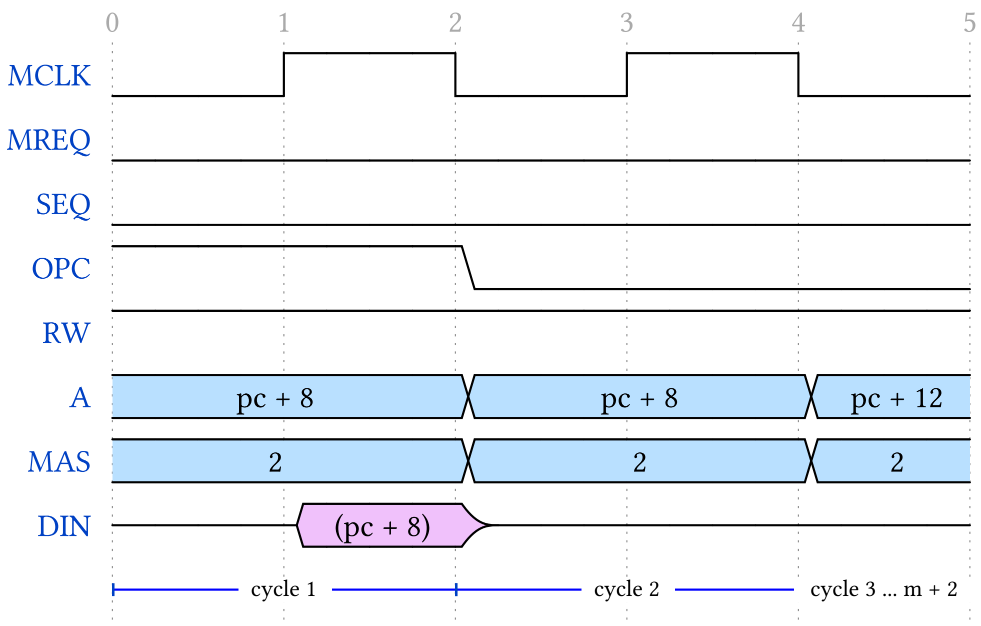


Figure 40: Multiply Accumlate Long Instruction Cycle

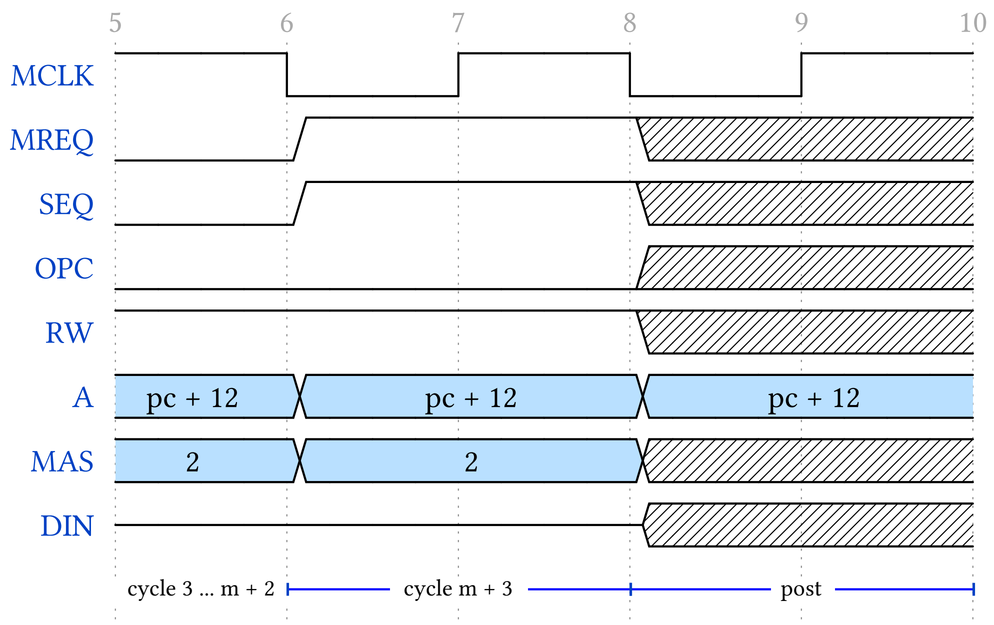


Figure 41: Multiply Accumulate Long Instruction Cycle (continued)

5.23. Load Register Instruction Cycle (LRI-Cycle)

Related Instructions
LDR , LDRB , LDRBT , LDRH , LDRSB , LDRSH , LDRT

Related Procedures
test_LRI_cycle
gba_request_LRI_cycle
memory_respond_LRI_cycle

Duration: 3 to 5 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
alu	The instruction operand—ie, the first source address.
i	MAS, 2 for ARM state, 1 for Thumb state.
s	MAS variable (BYTE , HALFWORD , or WORD).

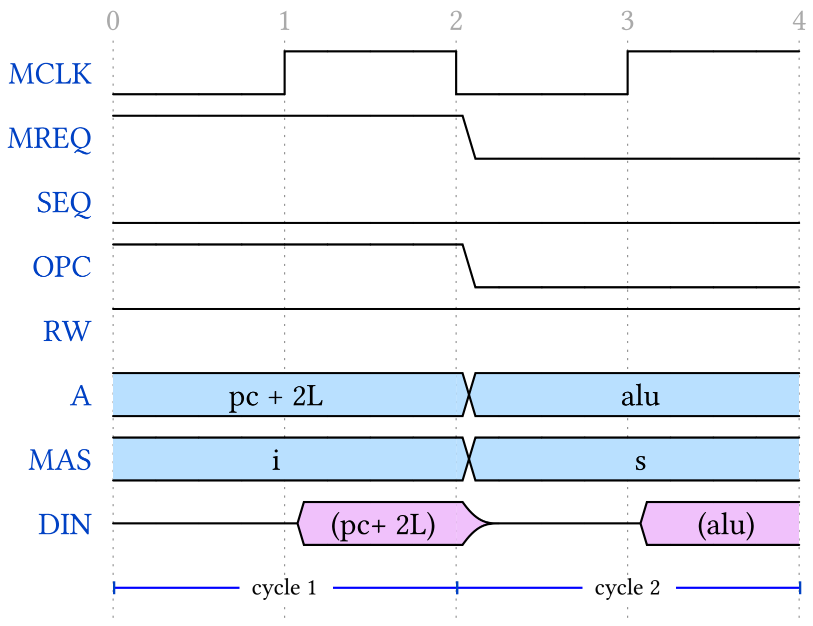


Figure 42: Load Register Instruction Cycle (normal)

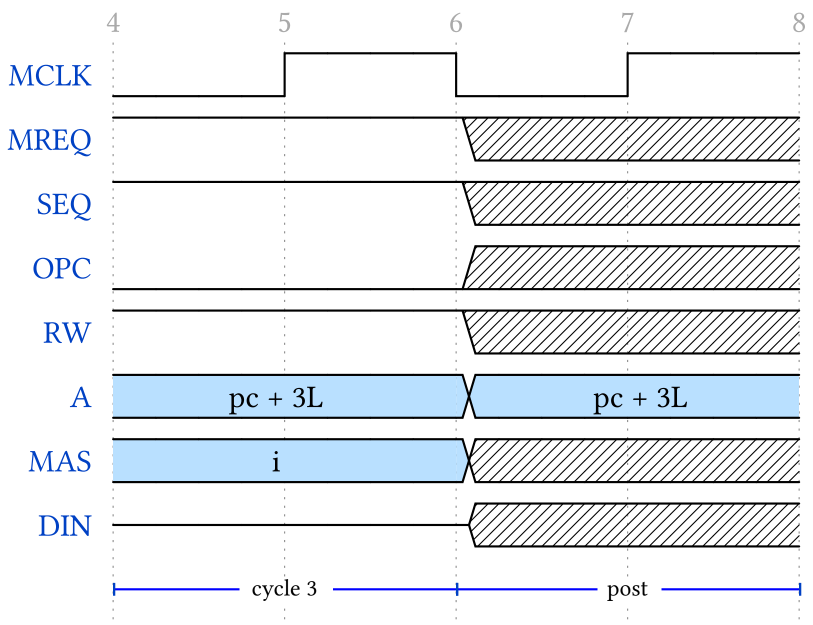


Figure 43: Load Register Instruction Cycle (normal) (continued)

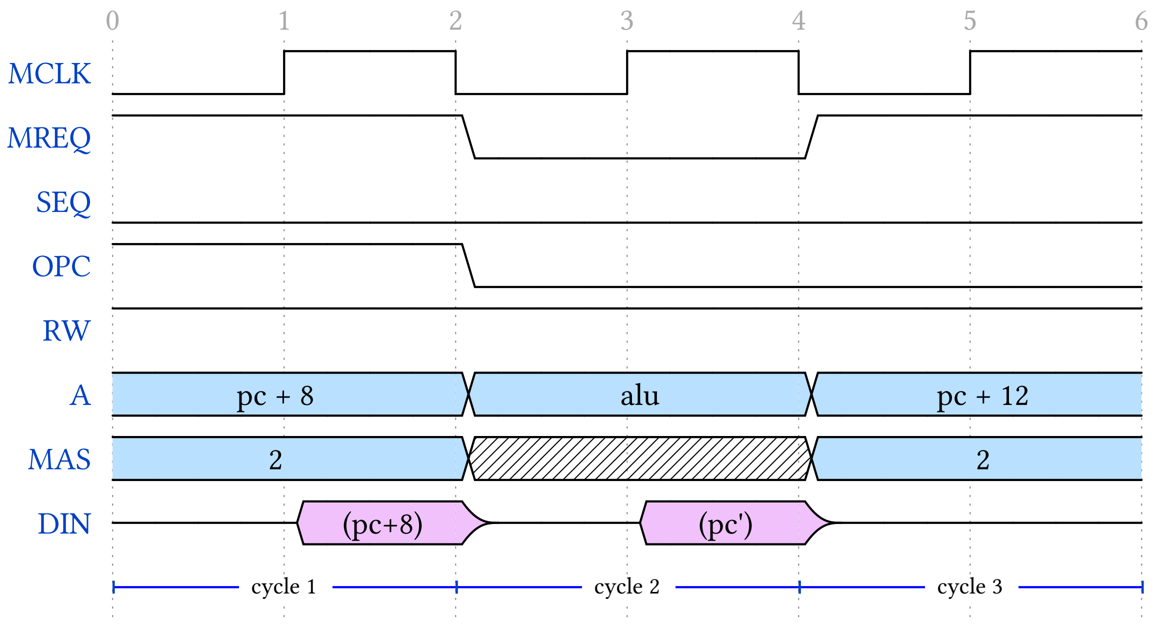


Figure 44: Load Register Instruction Cycle (dest=pc)

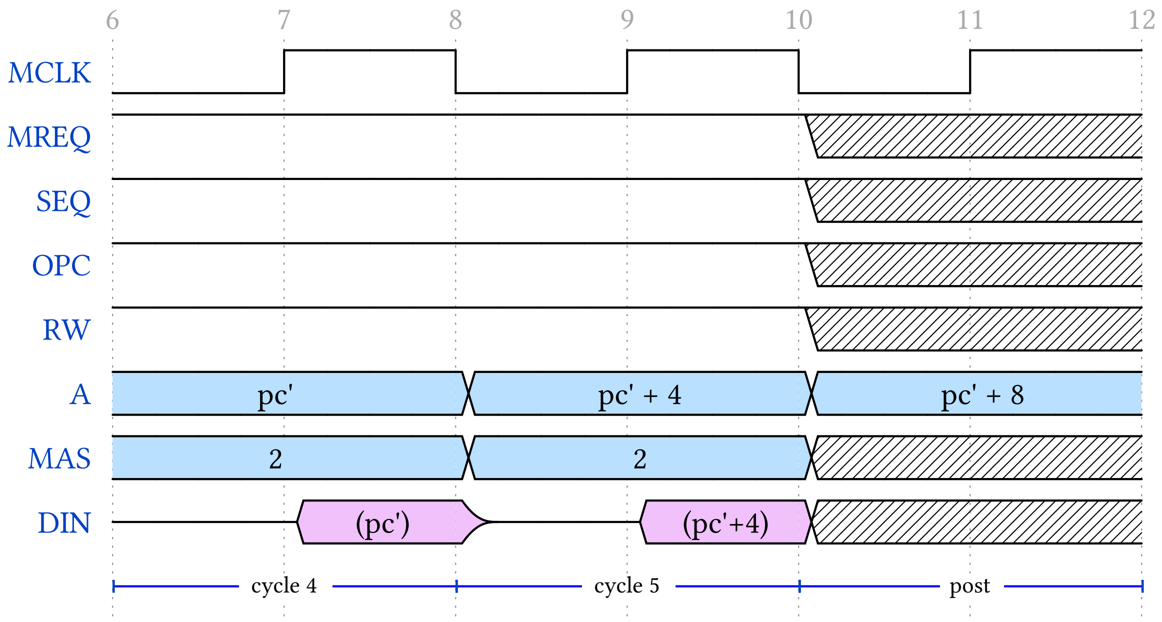


Figure 45: Load Register Instruction Cycle (dest=pc) (continued)

5.24. Store Register Instruction Cycle (SRI-Cycle)

Related Instructions
STR , STRB , STRBT , STRH , STRT

Related Procedures
test_SRI_cycle
gba_request_SRI_cycle
memory_respond_SRI_cycle

Duration: 2 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
alu	The instruction operand—ie, the first target address.
i	MAS, 2 for ARM state, 1 for Thumb state.
s	MAS variable (BYTE , HALFWORD , or WORD).

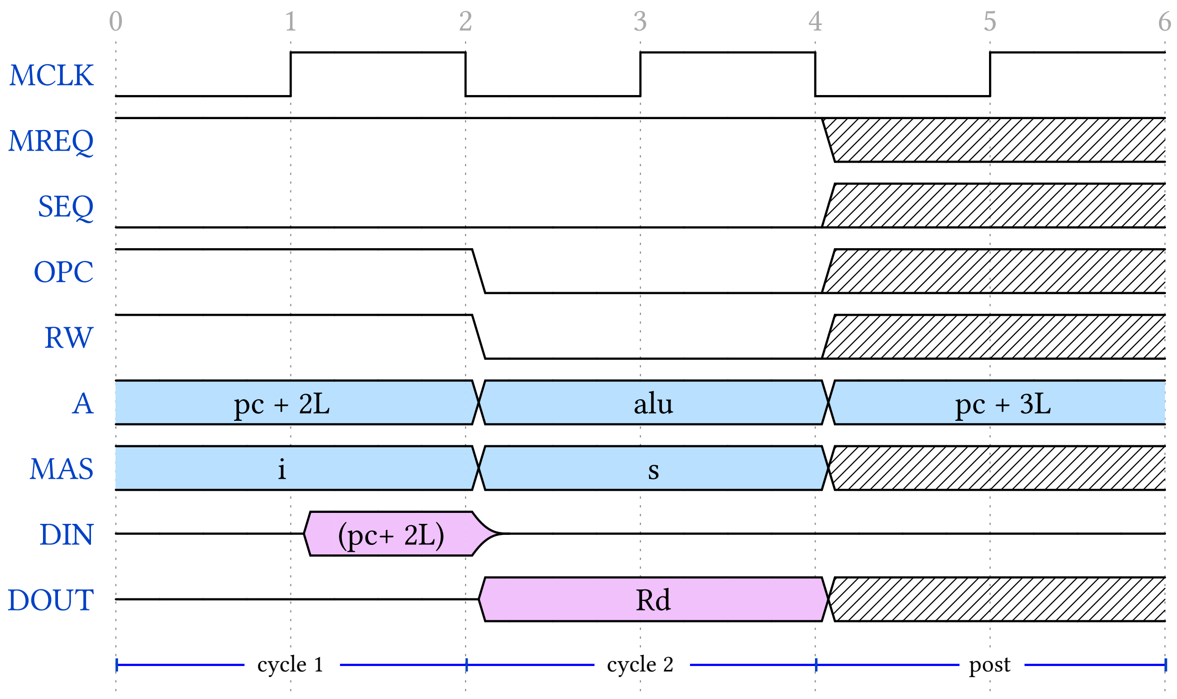


Figure 46: Store Register Instruction Cycle

5.25. Load Multiple Register Instruction Cycle (LMRI-Cycle)

Related Instructions
LDM

Related Procedures
test_LMRI_cycle
gba_request_LMRI_cycle
memory_respond_LMRI_cycle

Duration: 3 to 6 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
alu	The instruction operand—ie, the first source address.
i	MAS, 2 for ARM state, 1 for Thumb state.
s	MAS variable (BYTE , HALFWORD , or WORD).

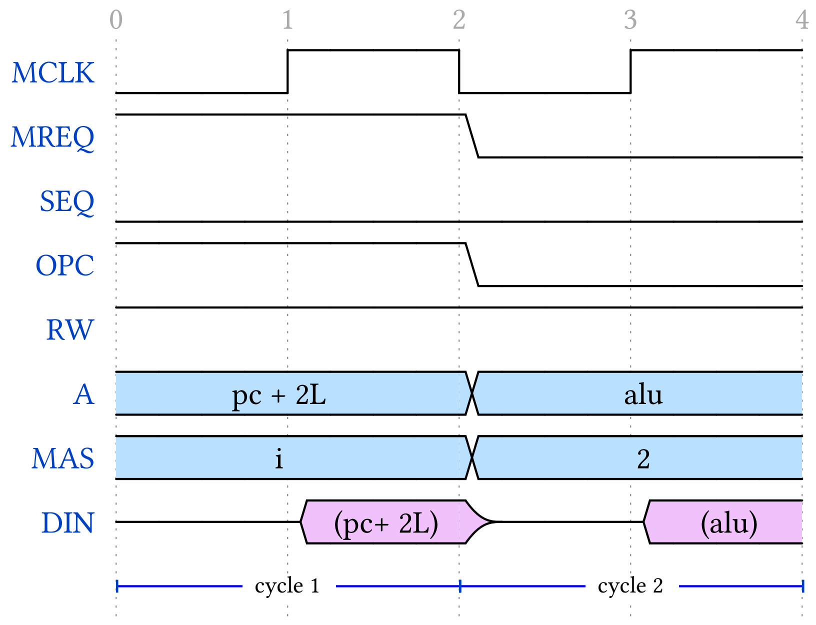


Figure 47: Load Multiple Register Instruction Cycle (single register)

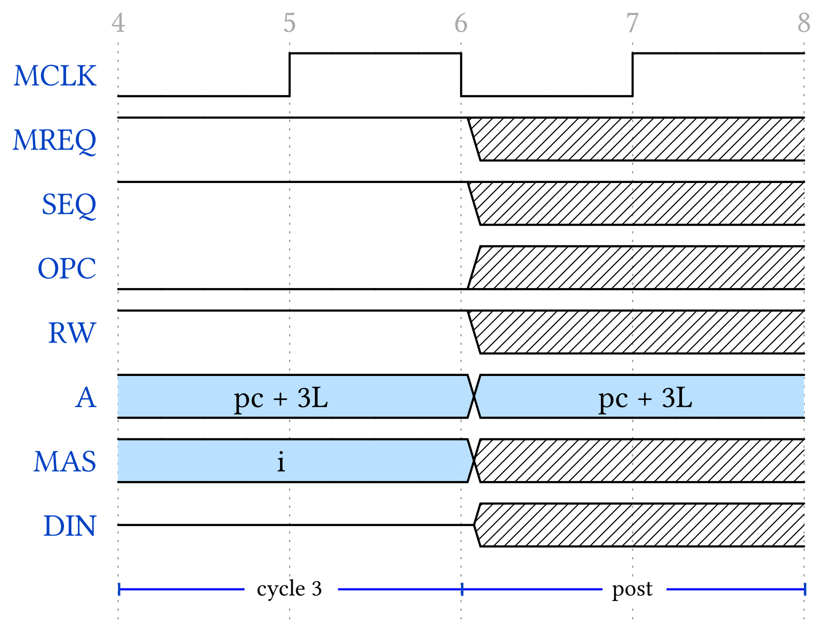


Figure 48: Load Multiple Register Instruction Cycle (single register) (continued)

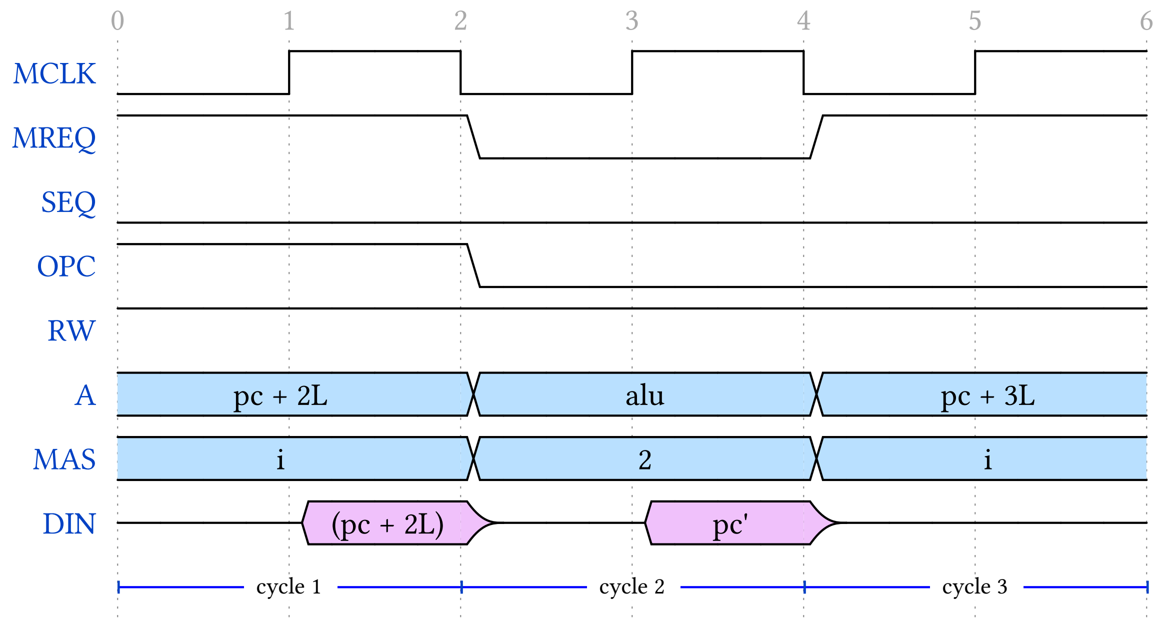


Figure 49: Load Multiple Register Instruction Cycle (single regiser dest=pc)

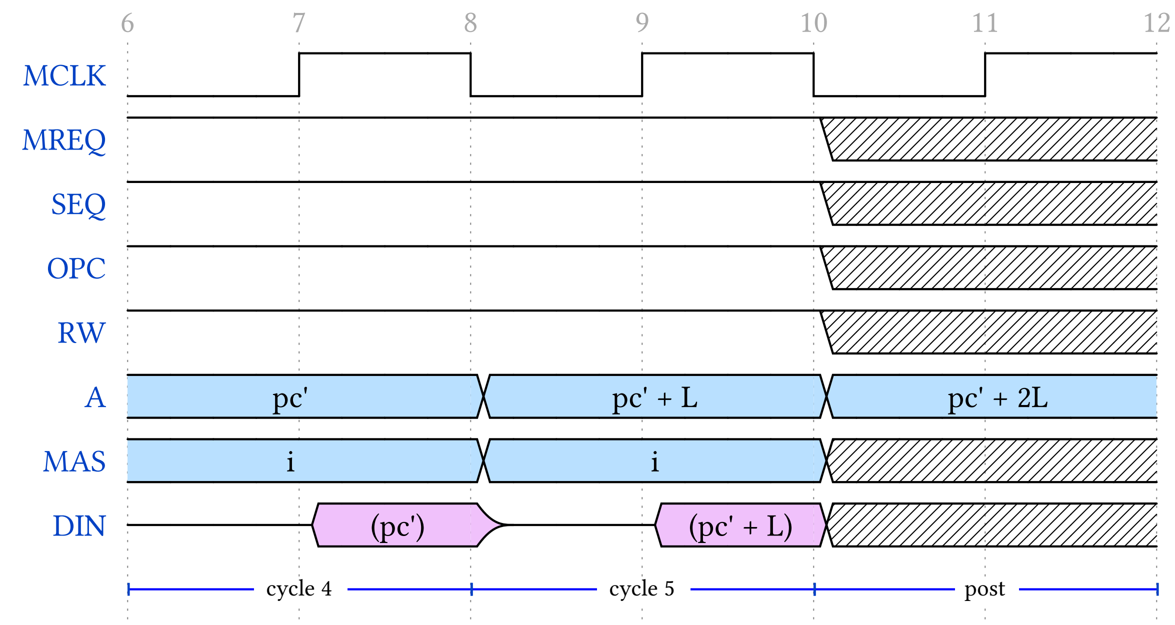


Figure 50: Load Multiple Register Instruction Cycle (single register dest=pc) (continued)

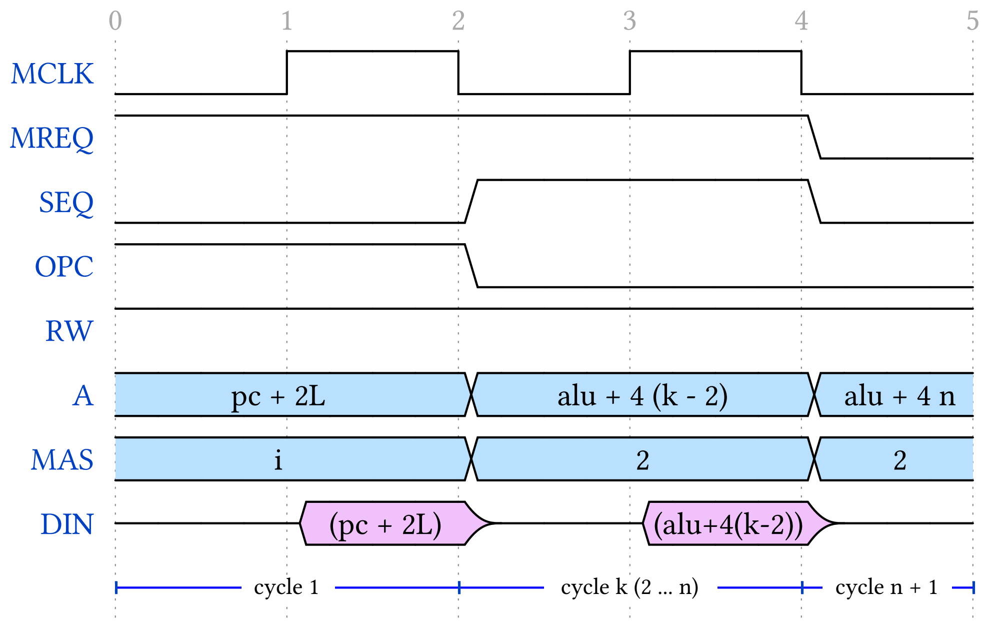


Figure 51: Load Multiple Register Instruction Cycle (n registers)

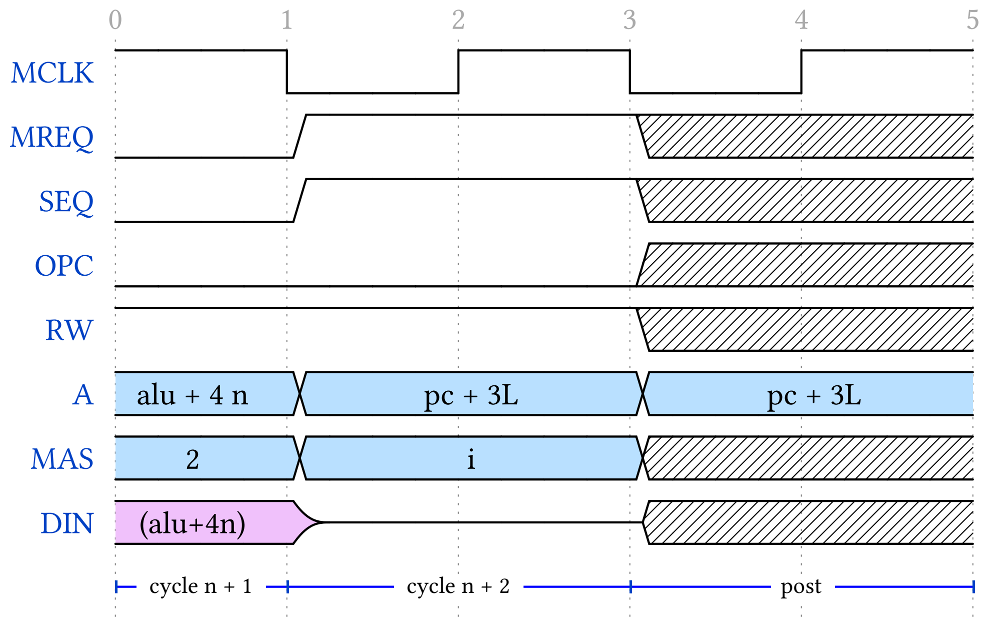


Figure 52: Load Multiple Register Instruction Cycle (n registers) (continued)

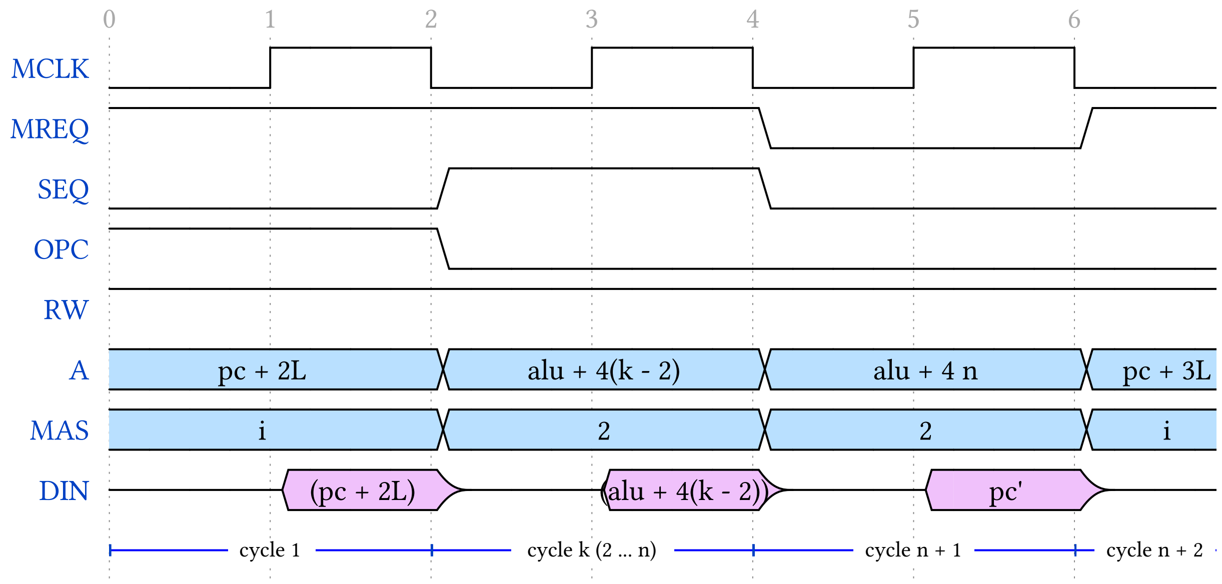


Figure 53: Load Multiple Register Instruction Cycle (n registers including pc)

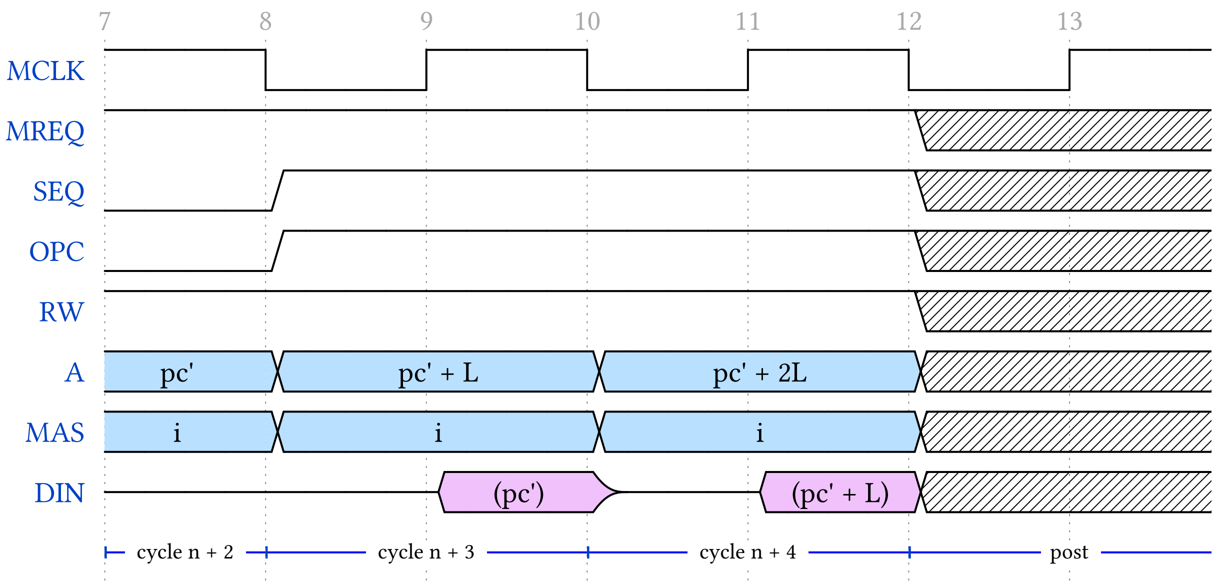


Figure 54: Load Multiple Register Instruction Cycle (n registers including pc) (continued)

5.26. Store Multiple Register Instruction Cycle (SMRI-Cycle)

Related Instructions
STM

Related Procedures
test_SMRI_cycle
gba_request_SMRI_cycle
memory_respond_SMRI_cycle

Duration: 2 to 3 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
alu	The instruction operand—ie, the first target address.
i	MAS, 2 for ARM state, 1 for Thumb state.
s	MAS variable (BYTE , HALFWORD , or WORD).
R[k]	The value in the *k*-th register.

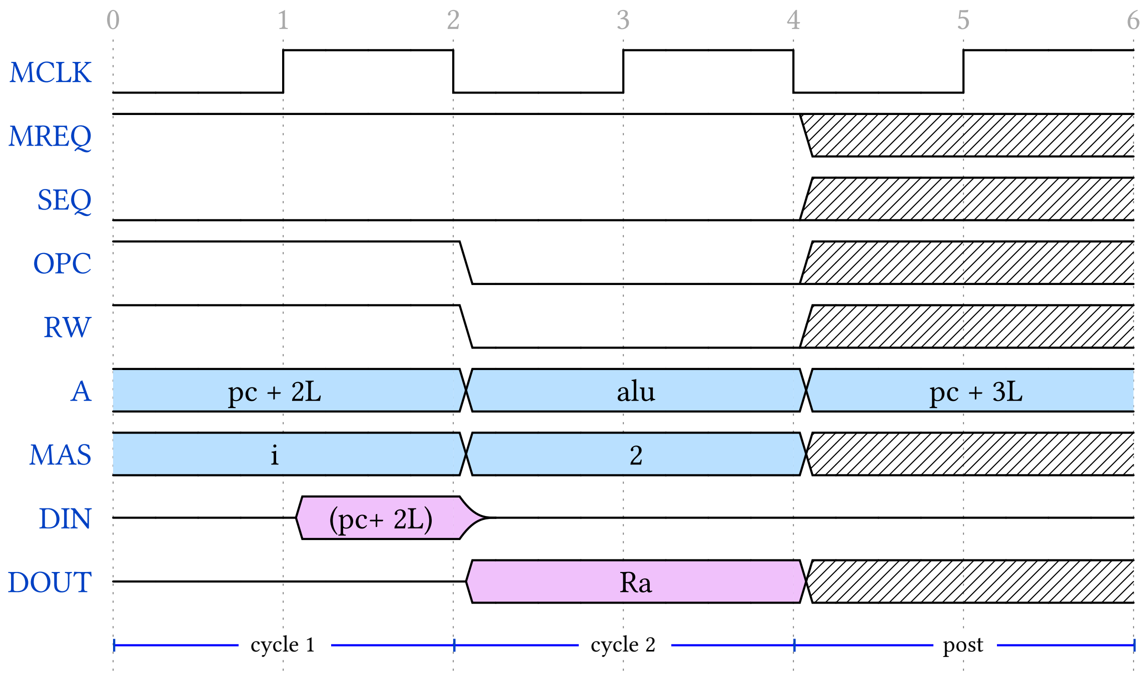


Figure 55: Store Multiple Register Instruction Cycle (single register)

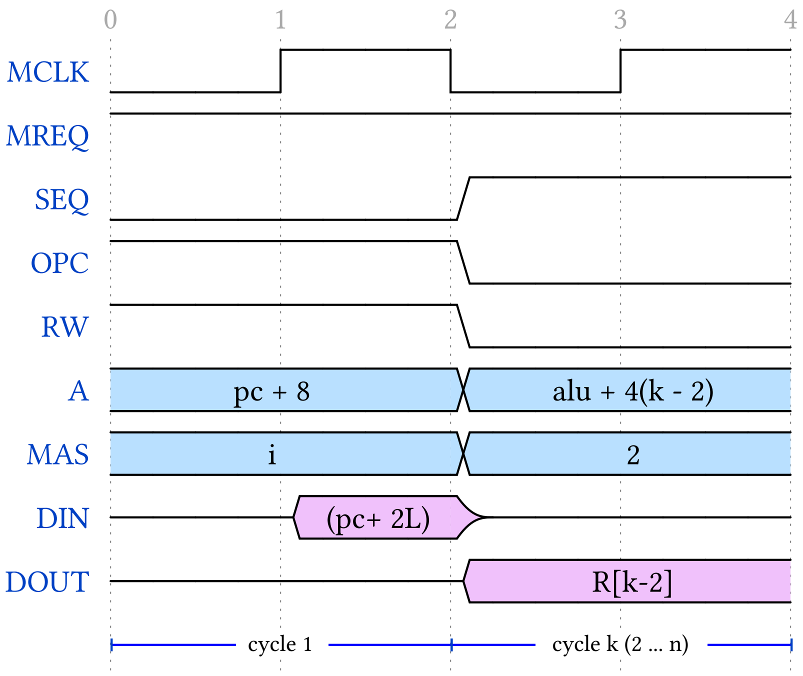


Figure 56: Store Multiple Register Instruction Cycle (n registers)

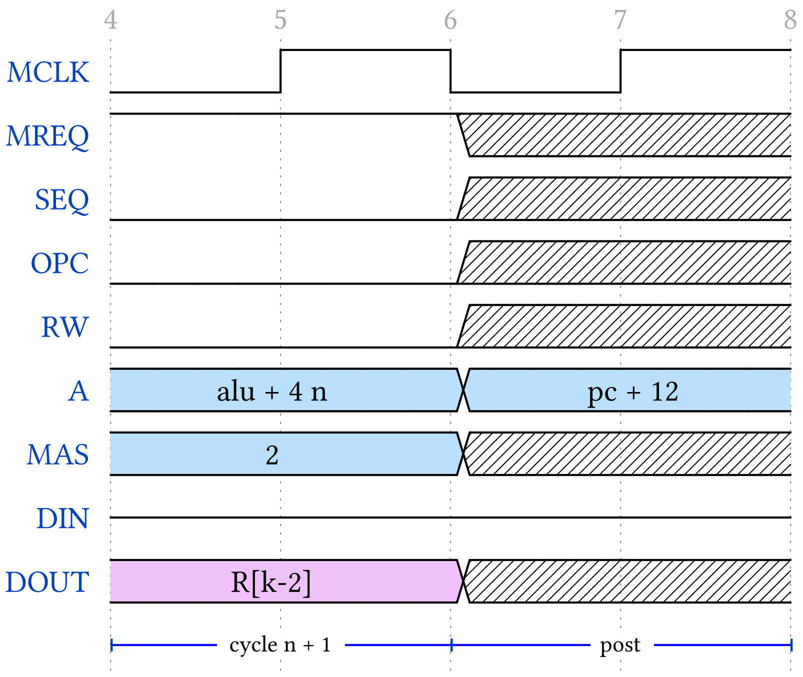


Figure 57: Store Multiple Register Instruction Cycle (n registers) (continued)

5.27. Data Swap Instruction Cycle (DSI-Cycle)

Related Instructions
SWP , SWPB

Related Procedures
test_DSI_cycle
gba_request_DSI_cycle
memory_respond_DSI_cycle

Duration: 4 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
alu	The instruction operand—ie, the first target address.
s	MAS variable (BYTE or WORD).
R[k]	The value in the *k*-th register.

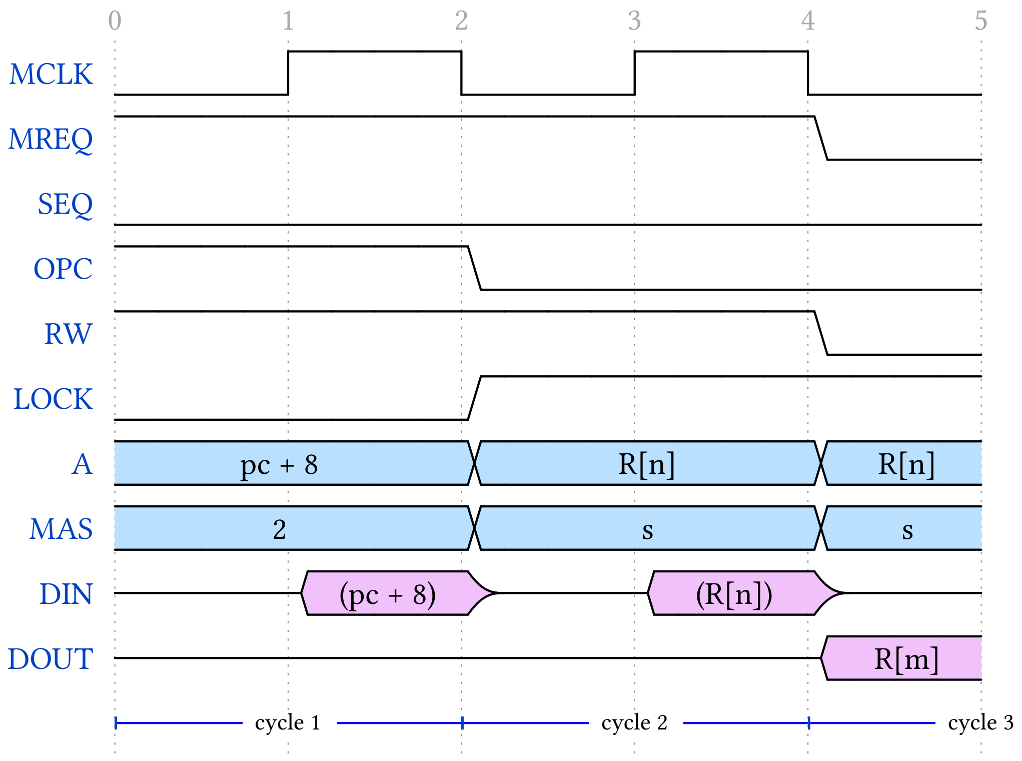


Figure 58: Data Swap Instruction Cycle

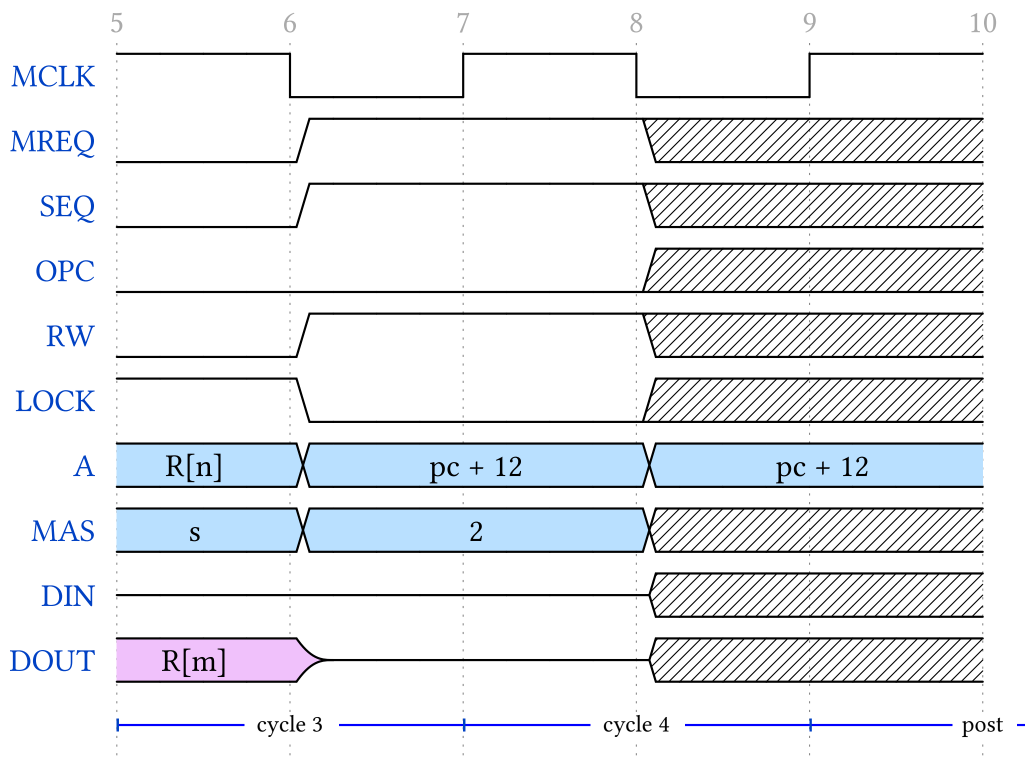


Figure 59: Data Swap Instruction Cycle (continued)

5.28. Software Interrupt and Exception Instruction Cycle (SIAEI-Cycle)

Related Instructions
SWI

Related Procedures
test_SIAEI_cycle
gba_request_SIAEI_cycle
memory_respond_SIAEI_cycle

Duration: 3 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
old	The processor mode, before executing the instruction
i	MAS, 2 for ARM state, 1 for Thumb state.
T	TBIT, before executing the instruction.
Xn	The exception address.

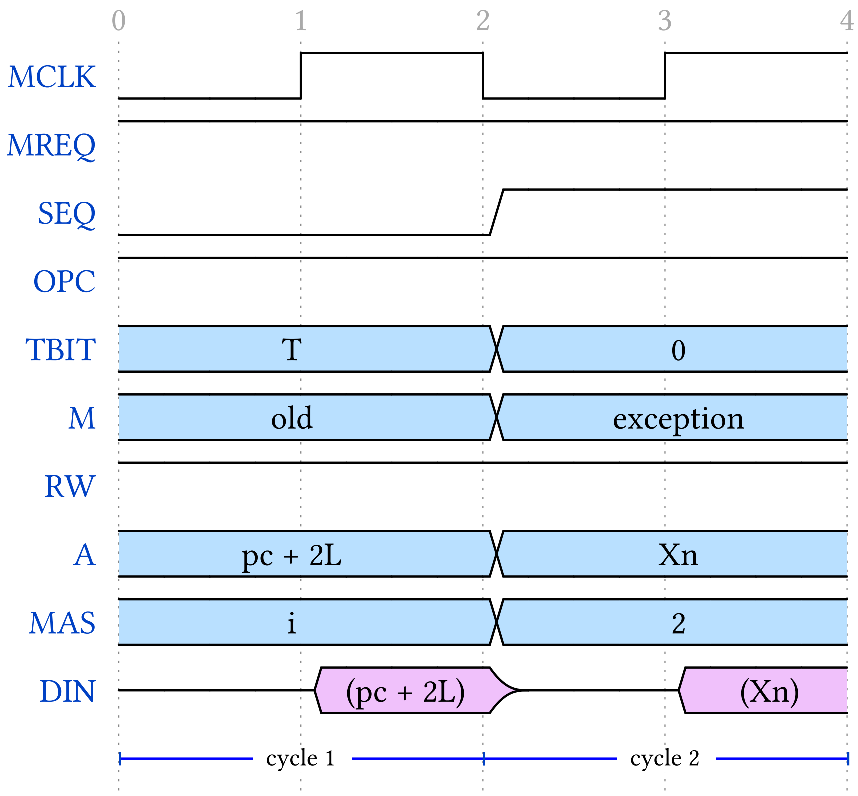


Figure 60: Software Interrupt and Exception Instruction Cycle

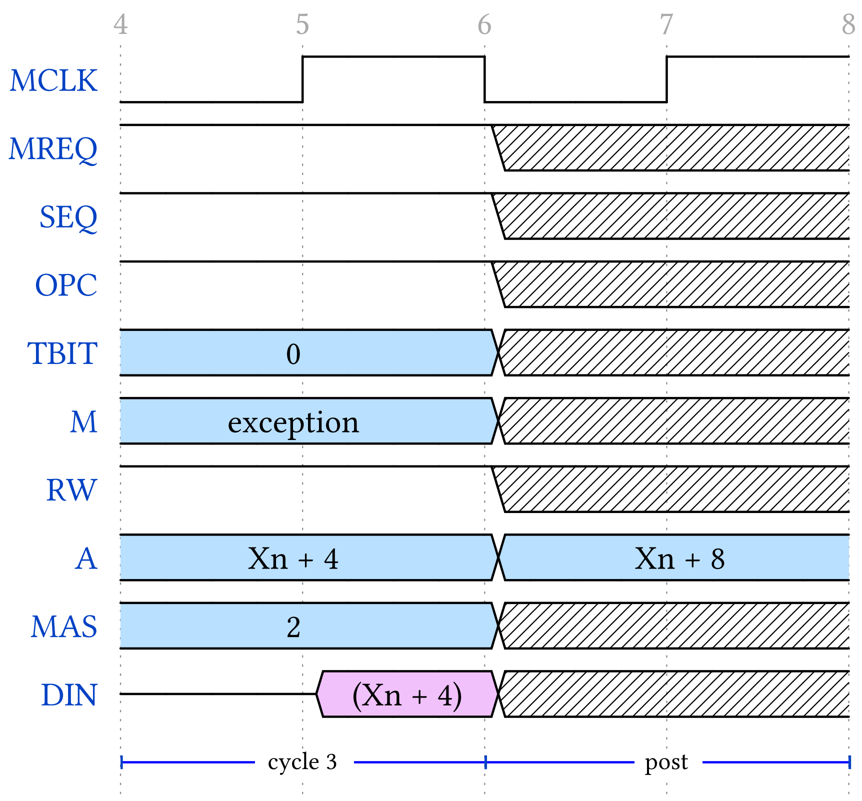


Figure 61: Software Interrupt and Exception Instruction Cycle (continued)

5.29. Undefined Instruction Cycle (UDI-Cycle)

Related Procedures
test_UDI_cycle
gba_request_UDI_cycle
memory_respond_UDI_cycle

Duration: 4 clock cycles

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
old	The processor mode, before executing the instruction
i	MAS, 2 for ARM state, 1 for Thumb state.
T	TBIT, before executing the instruction.
Xn	The exception address.

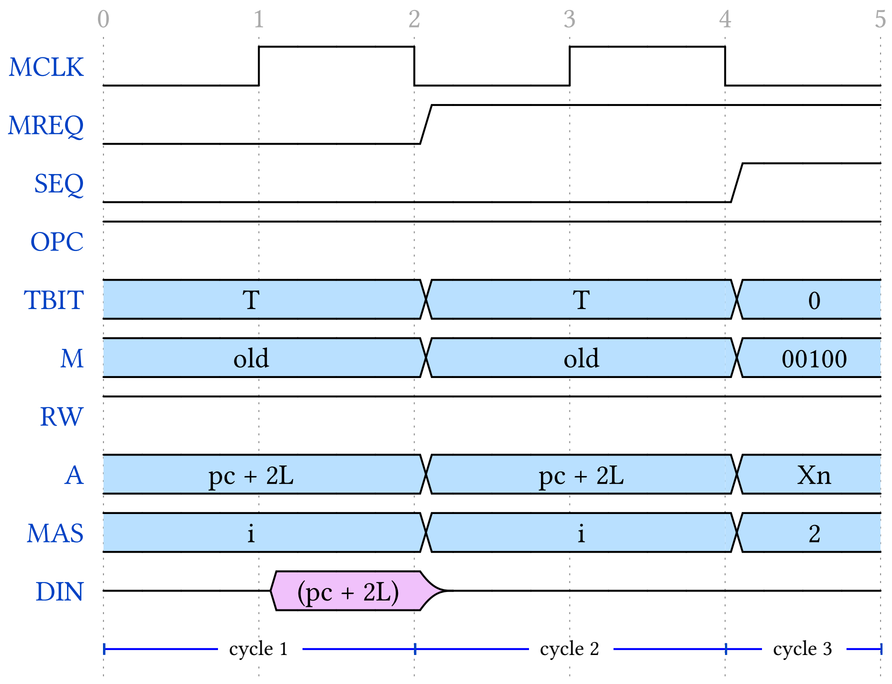


Figure 62: Undefined Instruction Cycle

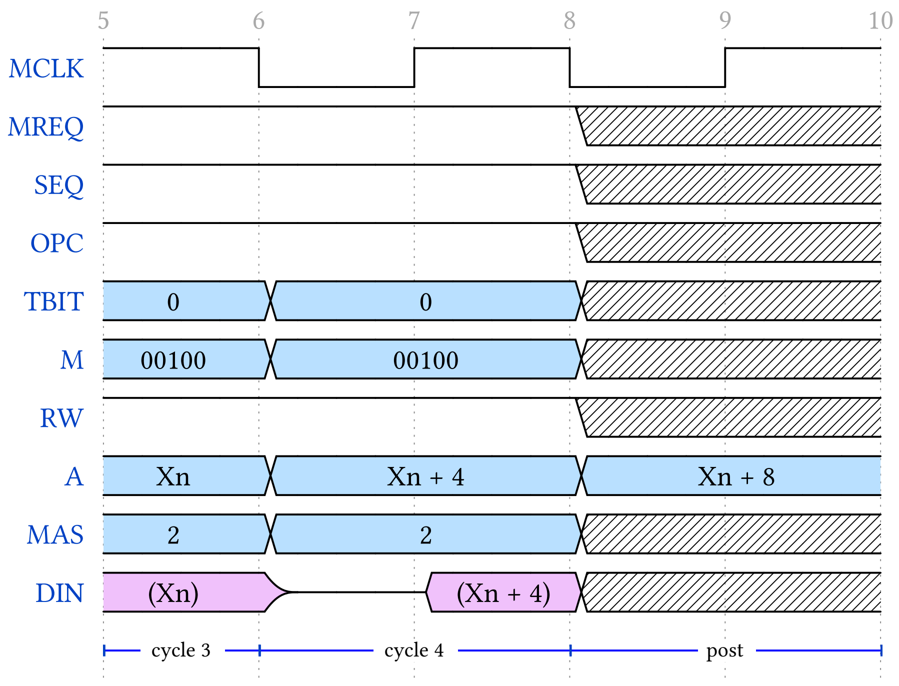


Figure 63: Undefined Instruction Cycle (continued)

5.30. Unexecuted Instruction Cycle (UEI-Cycle)

Related Procedures
test_UEI_cycle
gba_request_UEI_cycle
memory_respond_UEI_cycle

Duration: 1 clock cycle

	Parameters
pc	Program counter, before executing the instruction.
L	Instruction length, 4 for ARM state, 2 for Thumb state.
i	MAS, 2 for ARM state, 1 for Thumb state.

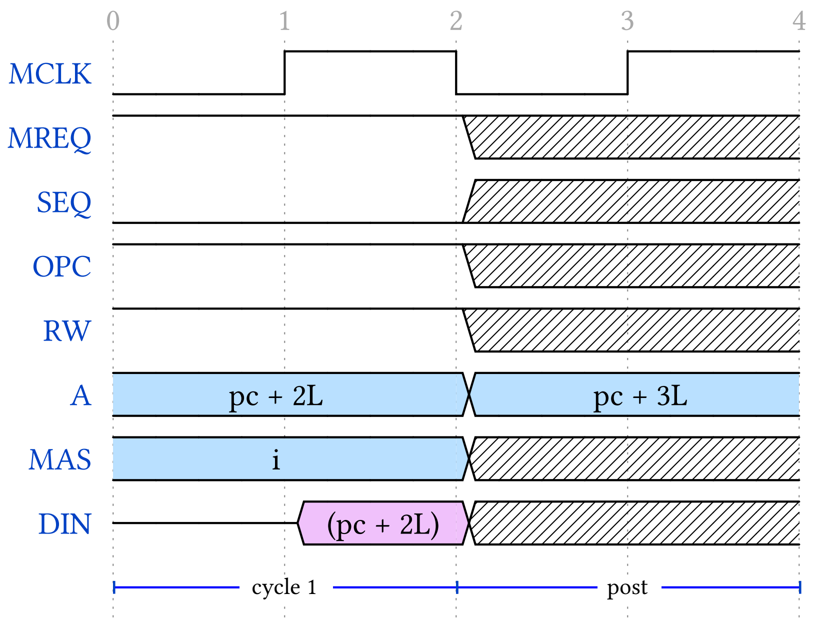


Figure 64: Unexecuted Instruction Cycle

Index of Figures