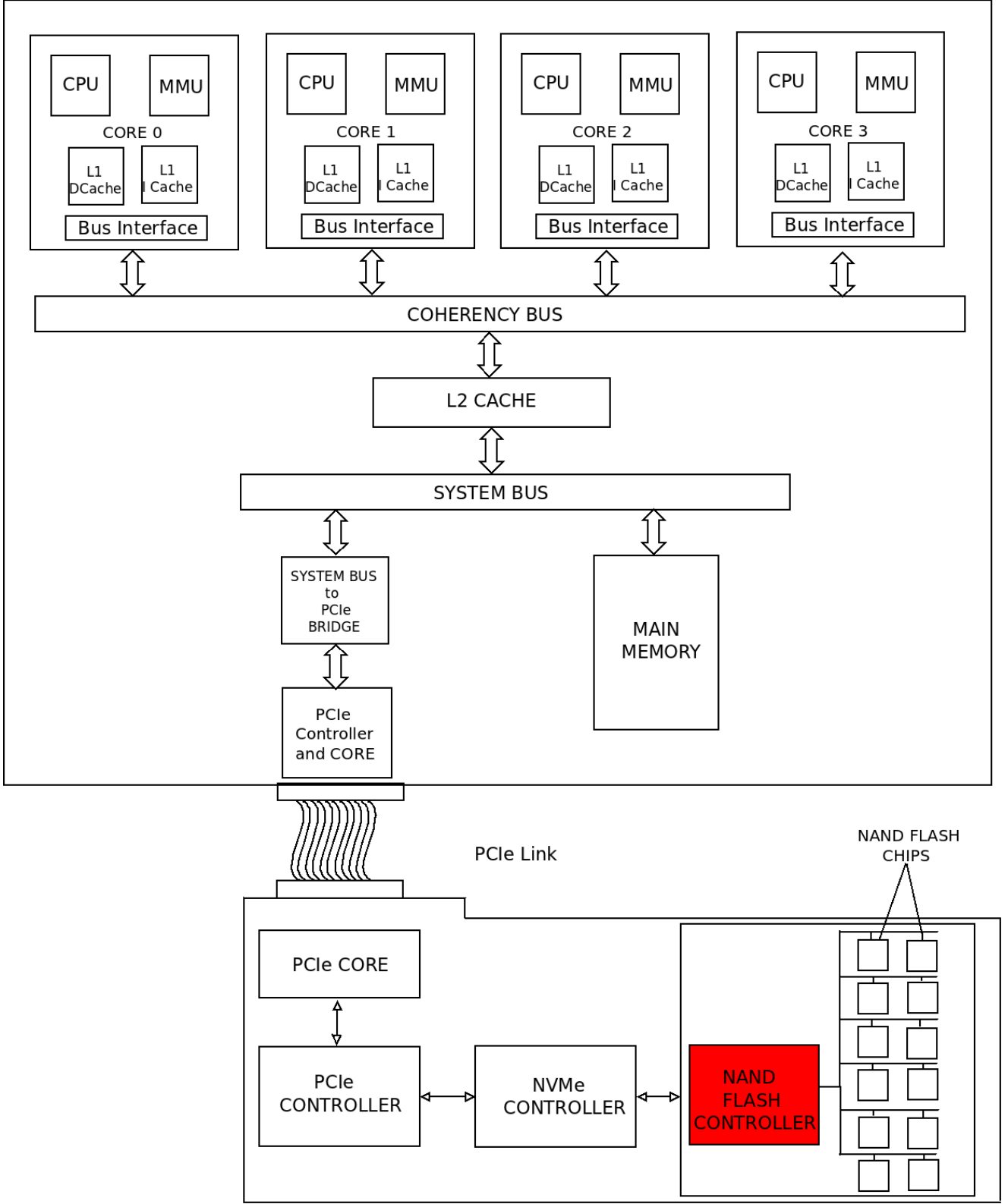
NAND FLASH CONTROLLER

AVINASH M

EE11M053

15TH MAY 2013

OVERALL ARCHITECTURE
&
MY CONTRIBUTION


CPU
MMU
CORE 0
L1 DCache
L1 I Cache
Bus Interface
CPU
MMU
CORE 1
L1 DCache
L1 I Cache
Bus Interface
CPU
MMU
CORE 2
L1 DCache
L1 I Cache
Bus Interface
CPU
MMU
CORE 3
L1 DCache
L1 I Cache
Bus Interface
COHERENCY BUS
L2 CACHE
SYSTEM BUS
SYSTEM BUS to PCIe BRIDGE
MAIN MEMORY
PCIe Controller and CORE
PCIe Link
NAND FLASH CHIPS
PCIe CORE
PCIe CONTROLLER
NVMe CONTROLLER
NAND FLASH CONTROLLER

INTRODUCTION TO BLUESPEC SYSTEM VERILOG

BSV LANGUAGE BACKGROUND

- High-level functional hardware description programming language.
- Leads to shorter, more abstract, and verifiable source code, as well as type-checked numeric code.
- Uses *rules* and *interface methods* for complex concurrency and control :
  - Across multiple shared resources
  - Across module boundaries

BSV CONSTRUCTS

- Rules are used to describe how data is moved from one state to another. Rules have two components – Rule Conditions and Rule Body.
- A module consists of three things: state, rules that operate on that state, and an interface to the outside world (surrounding hierarchy).
- Interfaces provide a means to group wires into bundles with specified uses, described by methods.
- Signals and buses are driven in and out of modules with methods.
- Functions are simply parameterized combinational circuits.

ADVANTAGES OF BSV

- Powerful parameterization.
- Much powerful “generate”.
- High-level of abstraction.
- Fully synthesizable at all levels of abstraction.
- Advanced clock management.
- Powerful static checking.
- Parameterized libraries of interconnect IP.
- Robustness to change.

FUNCTION LIBRARIES

- BSV has a large set of functional libraries.
- Data containers such as FIFO , registers , BRAMs etc .
- Circuits such as LFSR and completion buffer.
- Interface types , GET, PUT transactors.
- Multiple clockdomain Synchronizers .
- Various bus interfaces such as Common data bus , Z-bus etc.

BSV DESIGN & VERIFICATION FLOWS

- A BSV program may optionally include Verilog, SystemVerilog, VHDL, and C components.
- Design is simulated and debugged using source level simulation tool “*Bluesim*”.
- It can also be simulated by using the generated Verilog RTL.
- Simulation tools such as QuestaSim can directly be “*linked*” to Bluesim to debug the designs.
- Full visibility to Bluespec interfaces and state elements.

NEED FOR NAND FLASH

WHY NAND FLASH ?

- Non-volatile memory.
- High bandwidth & small access latency.
- Low power consumption.
- High chip density.
- Promising future of SSDs.
- Future memory -

  Hybrid Main Memory (NAND flash + DRAM).
  - 48% performance advantage over a disk cache configuration.
  - Cost effective way to build large main memory systems.

NAND FLASH VS. NOR FLASH

- The real benefits of NAND Flash are faster PROGRAM and ERASE times, as NAND Flash delivers sustained WRITE performance exceeding 7 MB/s.
- Block erase times are an impressive 500s for NAND Flash compared with 1 second for NOR Flash.
- NOR Flash requires approximately 44 I/O pins for a 16-bit device, while NAND Flash requires only 24 pins for a comparable interface.
- The one challenge is that NAND Flash is not well-suited for direct random access.

SSD VS. HDD

Characteristic	Hard Disk Drive	Solid State Drive
Optimal Transfer Size	Large Block (16KB+)	Small Block (4-8KB)
Optimal Workload	Sequential Read ***OR*** Write	Random Read ***AND*** Write
Performance	100's of IOPS	10,000's+ of IOPS
Available Capacity*	4TB (3.5" form factor)	1.6TB (2.5" form factor)
$/GB (Enterprise Grade)*	~ $0.10/GB (~ $400/4TB)	~ $1.60/GB (~ 750/480GB)

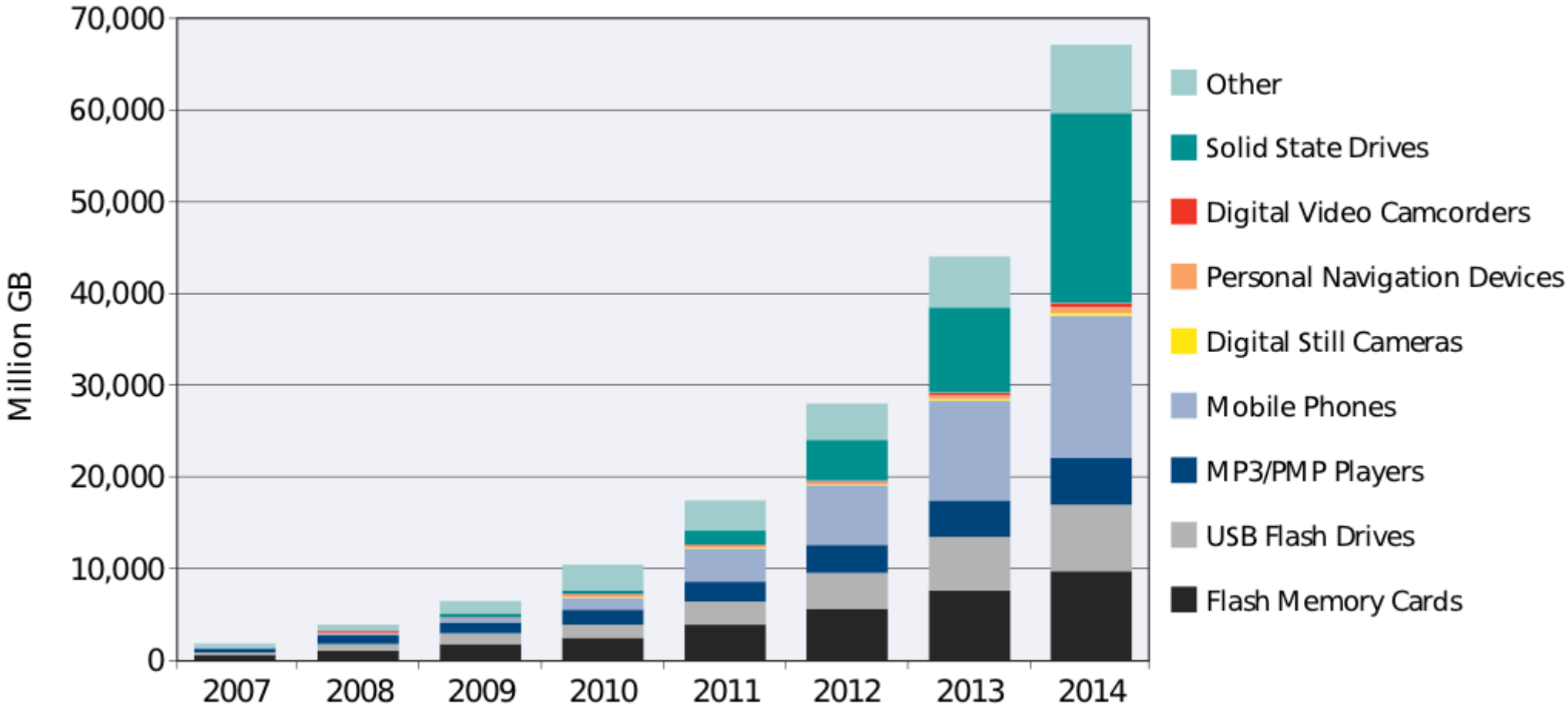
*As of October 2012

	Solid-State Drive	Rotating Hard Disk Drive
Shock	>1000 g	200 g
Vibration	16 g	1 g
Temperature	-40°C to +85°C	5°C to 55°C
Altitude	80,000 ft	< 50,000 ft
Power Consumption	0.2W	2.5W
MTBF	2-4M Hours	300K – 1M Hours
Duty Cycle	100%	20% - 40%

NAND FLASH APPLICATIONS

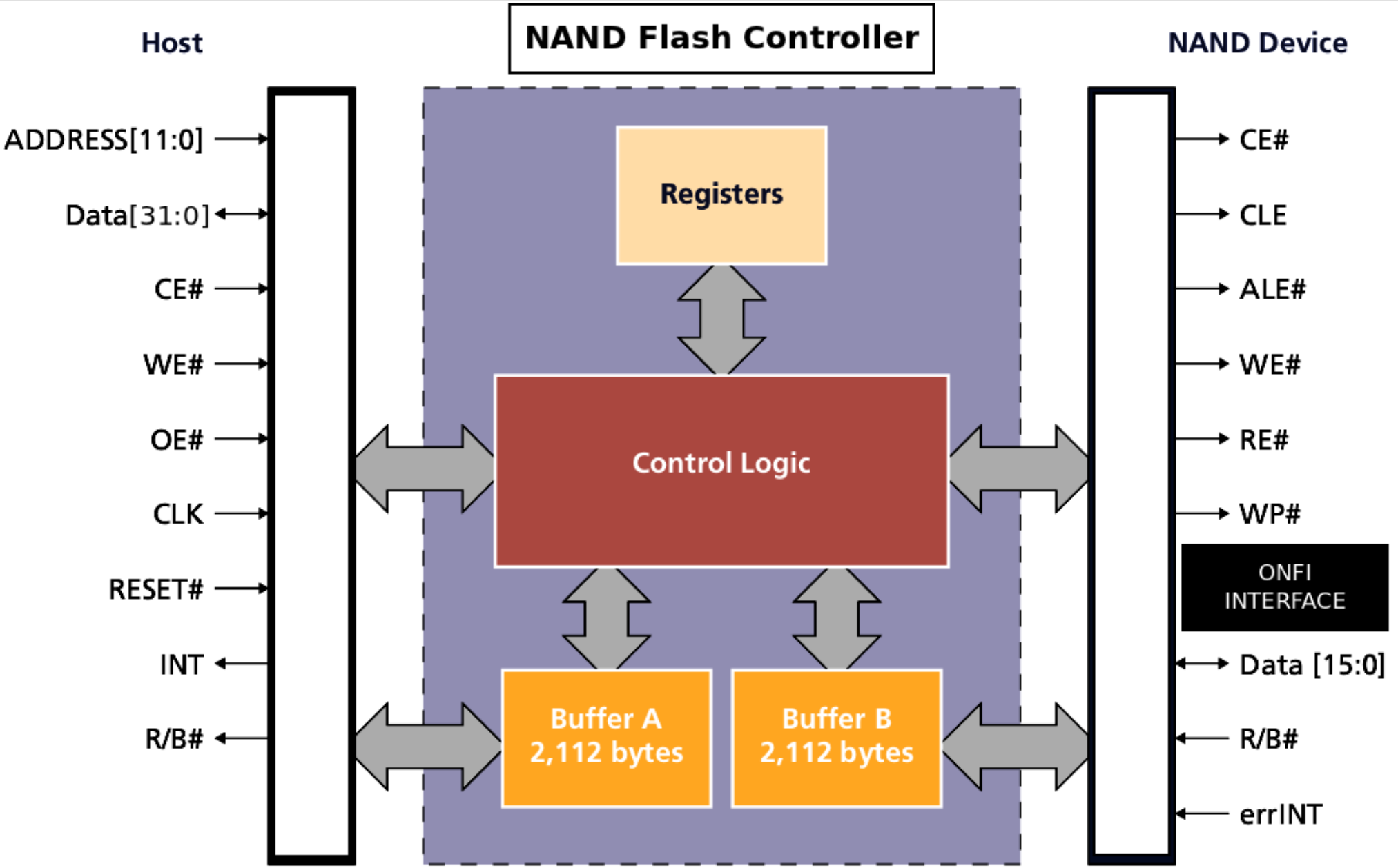
- USB flash drives
- Solid-state drives
- Memory cards
- Smart phones
- Ultrabooks
- Handheld audio & video players
- Digital cameras

NAND FLASH MARKET

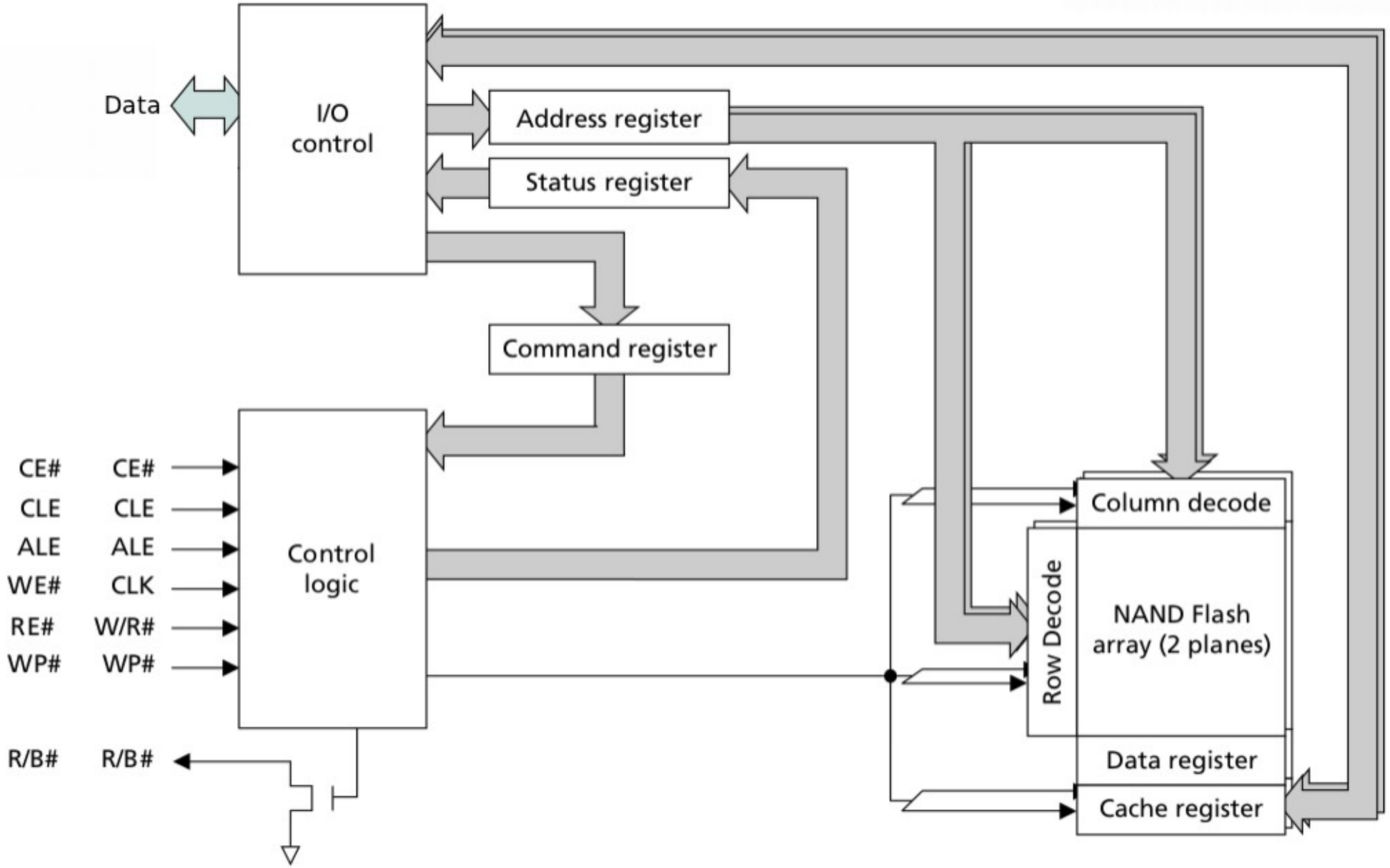


NAND FLASH CONTROLLER (NFC)

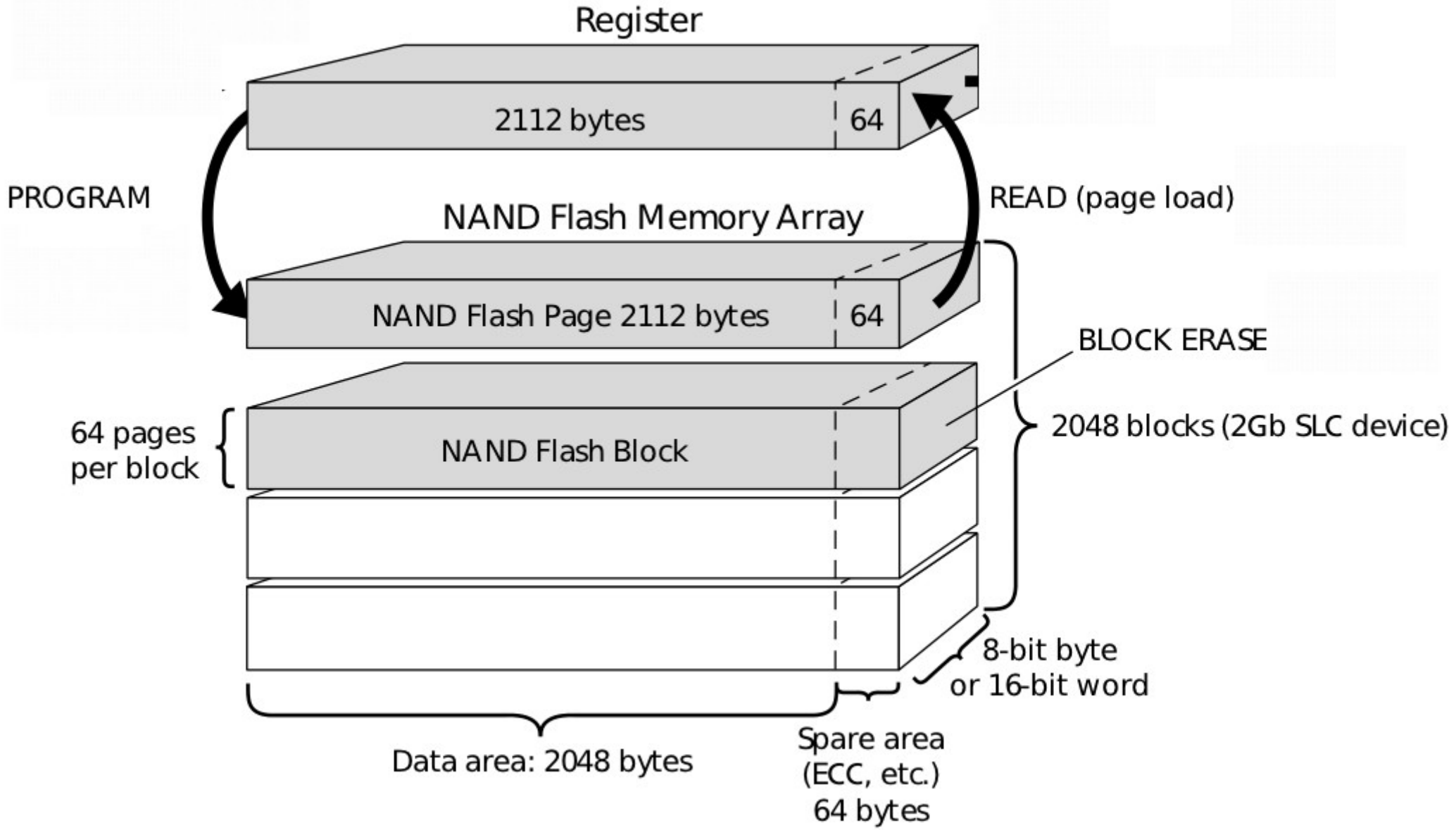
BLOCK DIAGRAM



FUNCTIONAL BLOCK DIAGRAM

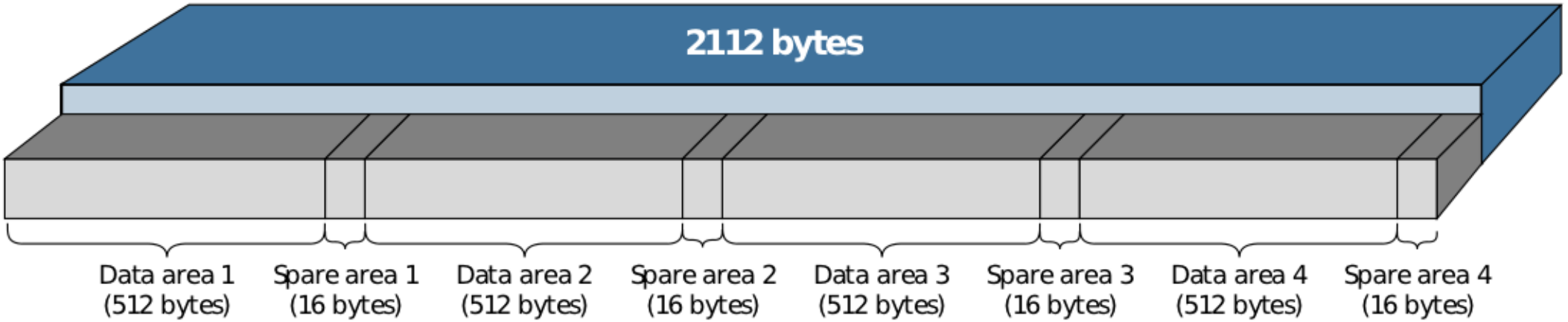


ORGANIZATION OF NAND FLASH

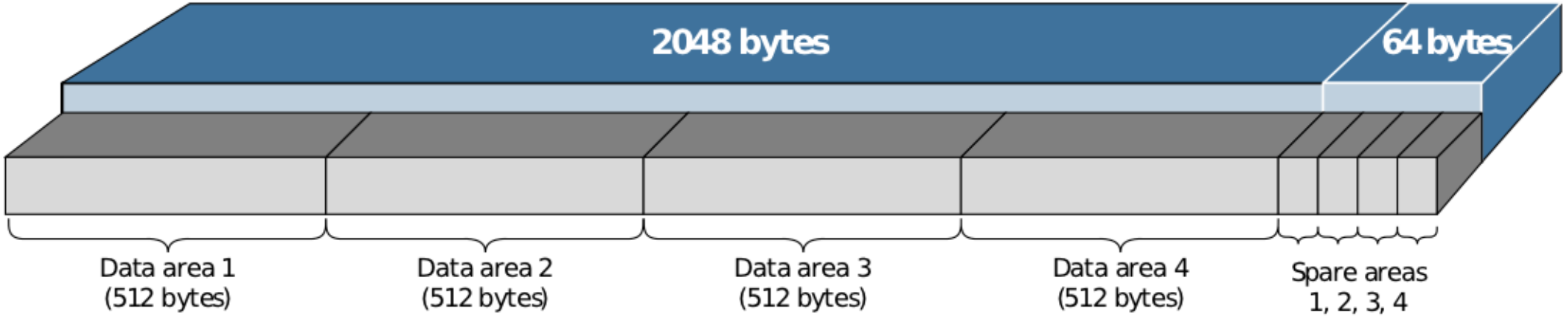


STORAGE METHODS

Adjacent Data and Spare Areas



Separate Data and Spare Areas



INTERFACE DESCRIPTION

HOST INTERFACE

SIGNAL	TYPE	DESCRIPTION
wr_address_from_nvm[11:0]	input	address input to the NFC from the host
wr_data_from_nvm[31:0]	input	data input to the NFC from the host
rg_data_to_nvm[31:0]	output	data output to the host from the NFC
wr_nand_ce_l	input (active low)	chip enable signal from the host, for NFC whether to accept input or not

HOST INTERFACE

SIGNAL	TYPE	DESCRIPTION
wr_nand_we_l	input (active low)	write enable allows data to be written by the host in the current buffer of NFC
wr_nand_oe_l	input (active low)	output enable allows data to be read by the host from NFC
wr_nand_reset_l	input (active low)	resets the NFC control logic
rg_interrupt	output	interrupt to host
rg_ready_busy_l	output (active low)	ready/busy status of NFC to the host

ONFi INTERFACE

SIGNAL	TYPE	DESCRIPTION
rg_data_to_flash[15:0]	output	data output to the target from the NFC
wr_data_from_flash[15:0]	input	data input from the target to the NFC
rg_onfi_ce_l	output (active low)	chip enable to accept input from NFC
rg_onfi_re_l	output (active low)	read enable allows data to be read from NAND flash
rg_onfi_we_l	output (active low)	write enable allows data to be written on NAND flash

ONFi INTERFACE

SIGNAL	TYPE	DESCRIPTION
rg_onfi_wp_l	output (active low)	write protect to prevent writing data to NAND flash
rg_onfi_cle_l	output (active low)	command latch enable provides command input to the target via data bus
rg_onfi_ale_l	output (active low)	address latch enable provides address input to the target via data bus
wr_interrupt	input	interrupt from target
wr_ready_busy_l	input (active low)	ready/busy status of the NAND flash to the NFC

MODULES
&
STATE MACHINES

MODULES & STATE MACHINES

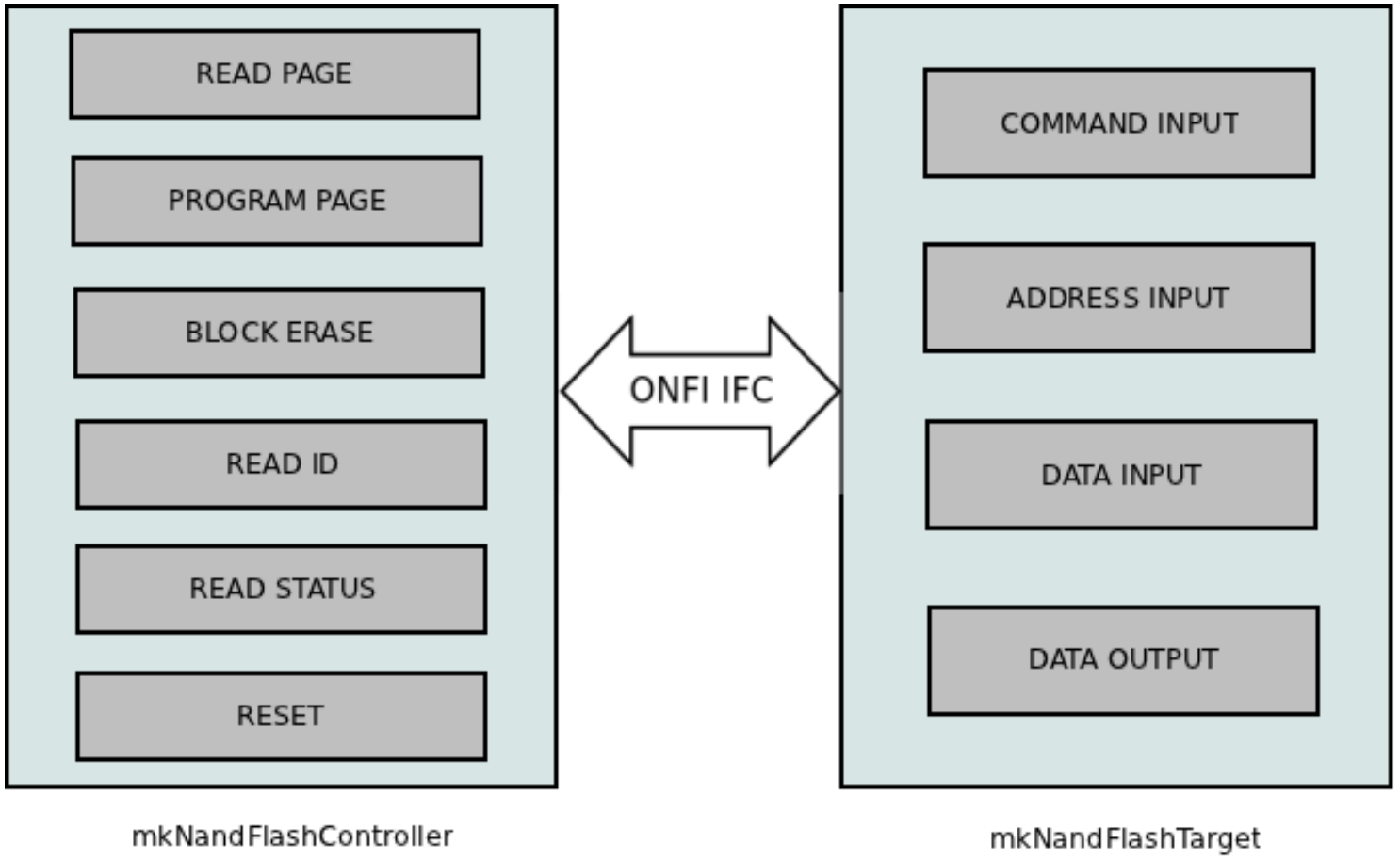
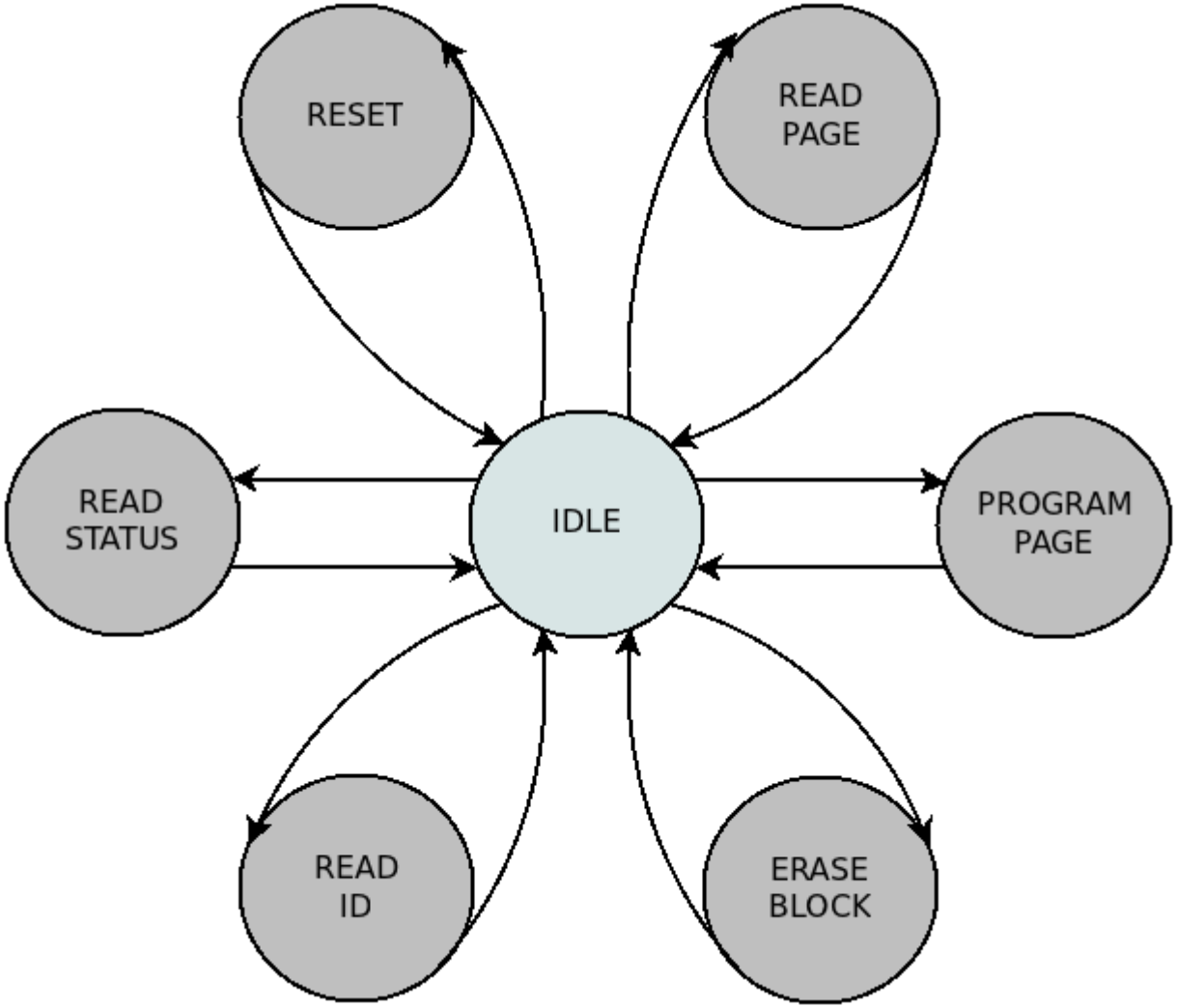


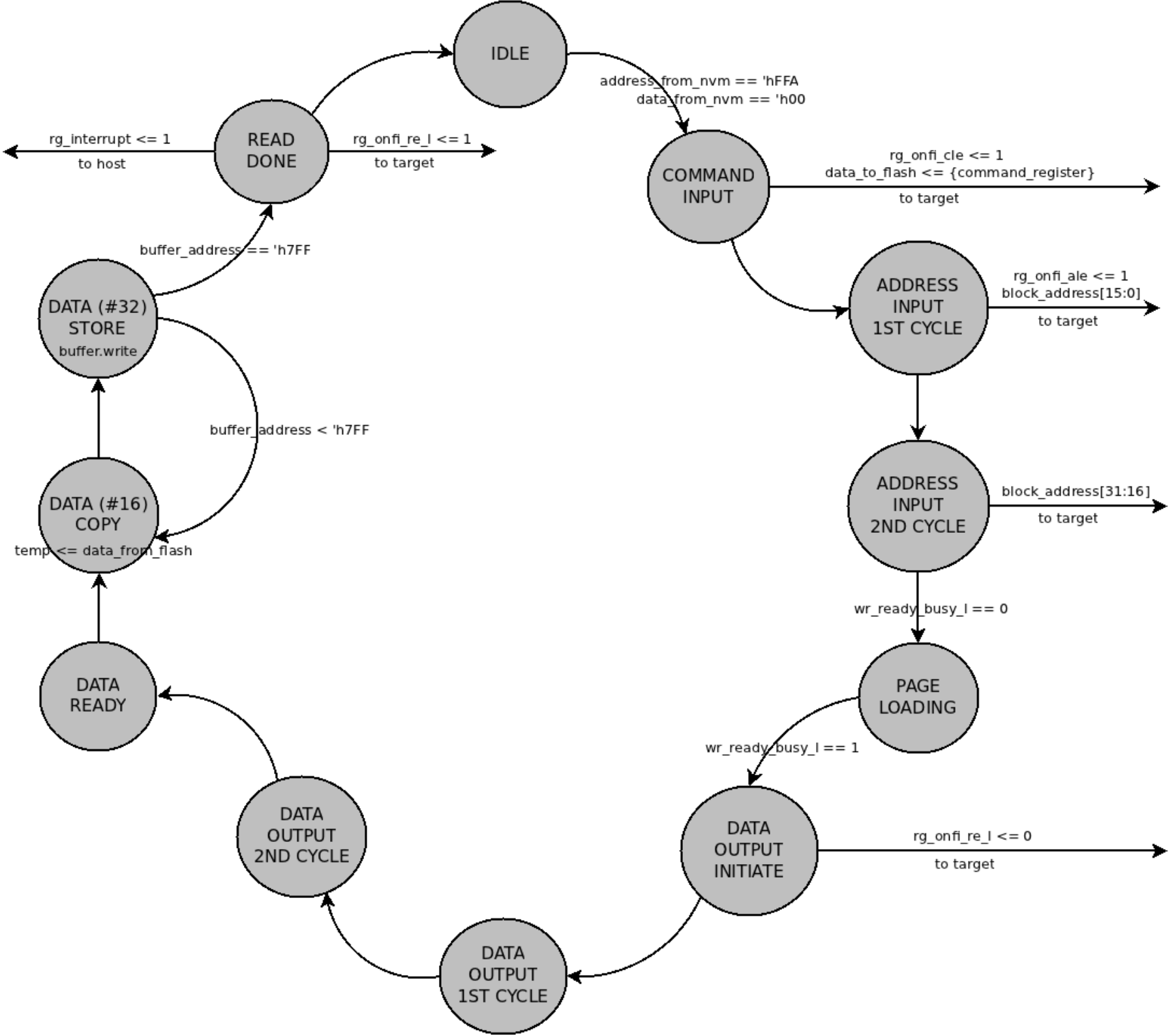
Fig : Modules & respective state machines used to implement them

STATE MACHINE OF NFC



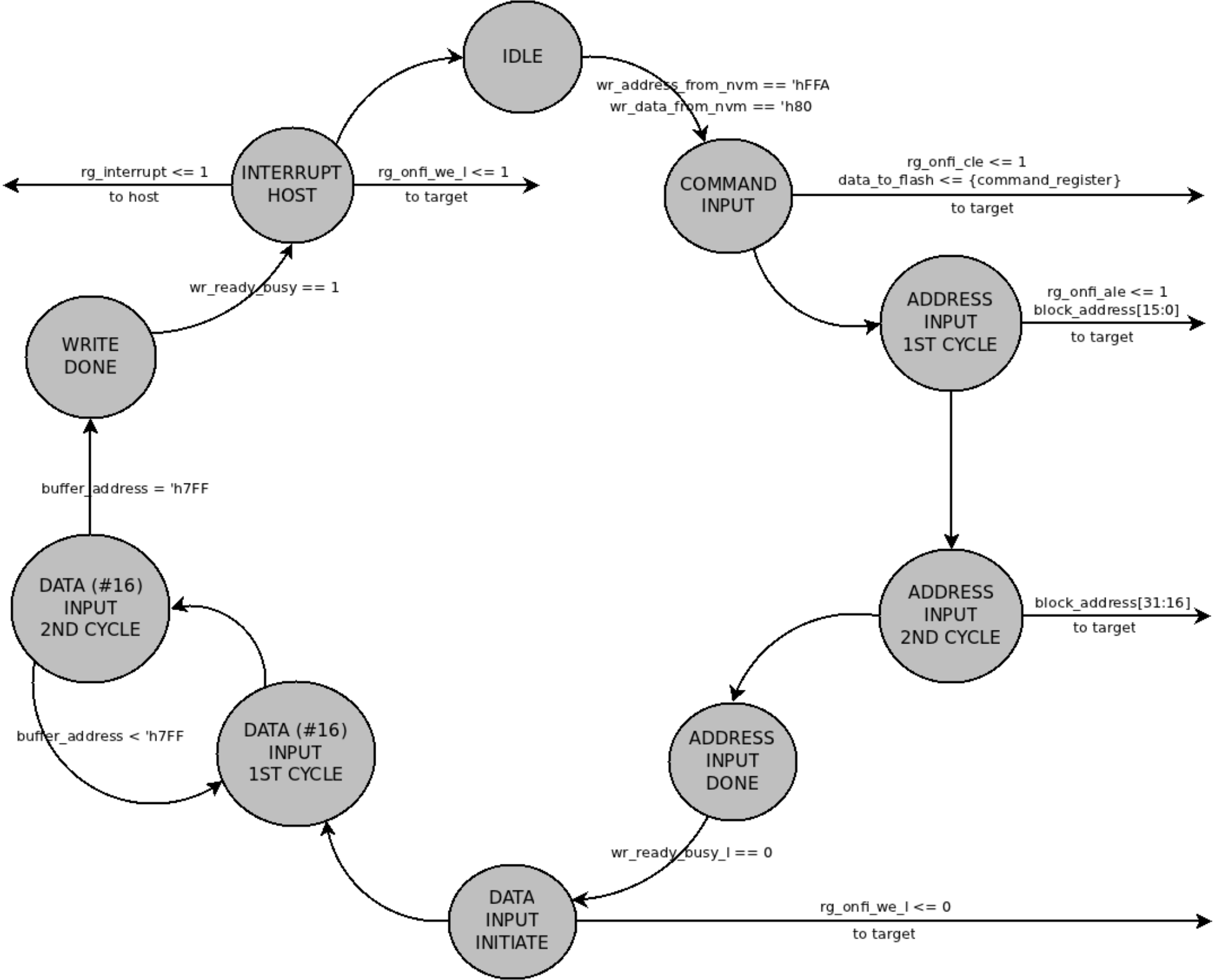
READ PAGE

- READ PAGE is initiated when the NAND Flash Controller receives the command ’00h’ from the Host.
- The Host sends the second and final command of Read Page i.e., ’30h’ after loading the address onto the *address register.*
- The NAND Flash Target makes the ready/busy signal go low indicating that it is busy copying the page from the address location to the data register of the Target, from which the NFC can read the data.
- The NAND Flash Controller sends an interrupt to the Host indicating that the Read Page process is complete, terminates the data output cycle and goes back to idle state.


IDLE
address_from_nvm == 'hFFA
data_from_nvm == 'h00
COMMAND INPUT
rg_onfi_cle <= 1
data_to_flash <= {command_register}
to target
ADDRESS INPUT 1ST CYCLE
rg_onfi_ale <= 1
block_address[15:0]
to target
ADDRESS INPUT 2ND CYCLE
block_address[31:16]
to target
wr_ready_busy_l == 0
PAGE LOADING
wr_ready_busy_l == 1
DATA OUTPUT INITIATE
rg_onfi_re_l <= 0
to target
DATA OUTPUT 1ST CYCLE
DATA OUTPUT 2ND CYCLE
DATA READY
DATA (#16) COPY
temp <= data_from_flash
buffer_address < 'h7FF
DATA (#32) STORE
buffer.write
buffer_address == 'h7FF
READ DONE
rg_interrupt <= 1
to host
rg_onfi_re_l <= 1
to target

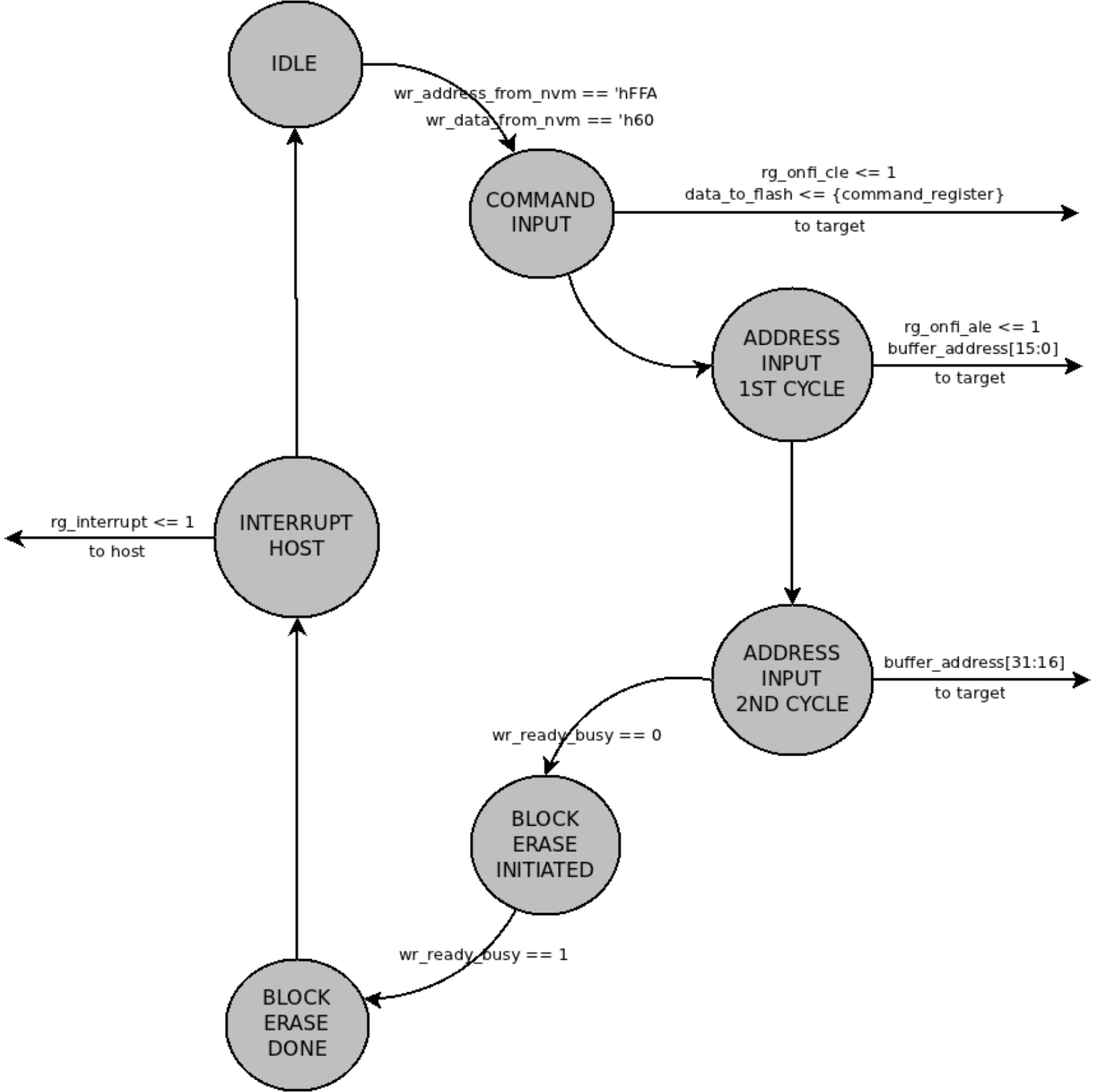
PROGRAM PAGE

- PROGRAM PAGE is initiated when the NAND Flash Controller receives the command ’80h’ from the Host.
- The Host sends the second and final command of Program Page i.e., ’10h’ after loading the address onto the *address register* and data onto the *data buffer(s).*
- NFC initiates the data input cycle by making the write enable(active low) signal go low and starts copying the data from the data buffer(s) to the NAND Flash Target.
- The NAND Flash Controller sends an interrupt to the Host indicating that the Program Page process is complete, terminates the data input cycle and goes back to idle state.


IDLE
wr_address_from_nvm == 'hFFA
wr_data_from_nvm == 'h80
COMMAND INPUT
rg_onfi_cle <= 1
data_to_flash <= {command_register}
to target
ADDRESS INPUT 1ST CYCLE
rg_onfi_ale <= 1
block_address[15:0]
to target
ADDRESS INPUT 2ND CYCLE
block_address[31:16]
to target
ADDRESS INPUT DONE
wr_ready_busy_l == 0
DATA INPUT INITIATE
rg_onfi_we_l <= 0
to target
DATA (#16) INPUT 1ST CYCLE
buffer_address < 'h7FF
DATA (#16) INPUT 2ND CYCLE
buffer_address = 'h7FF
WRITE DONE
wr_ready_busy == 1
INTERRUPT HOST
rg_interrupt <= 1
to host
rg_onfi_we_l <= 1
to target

BLOCK ERASE

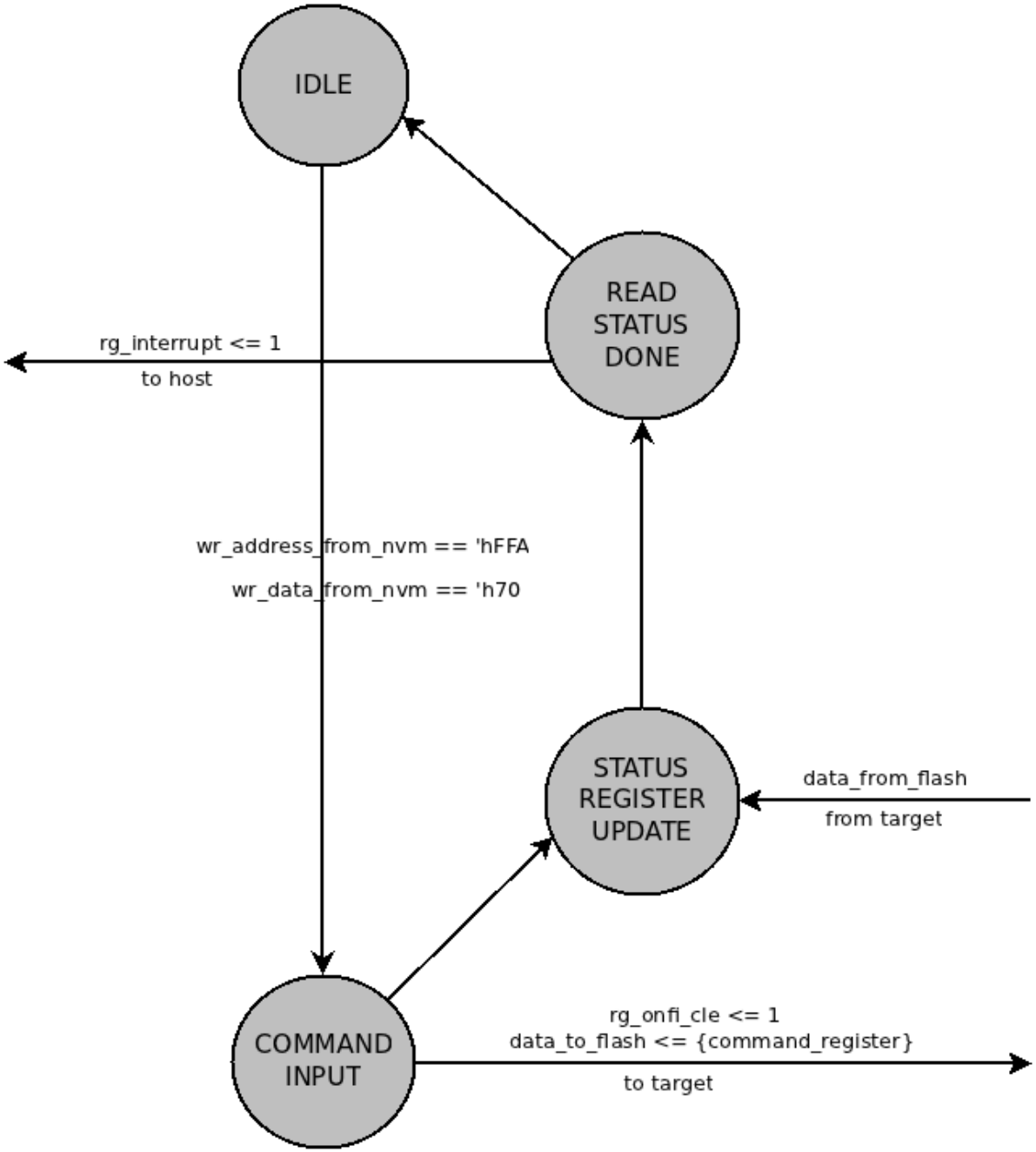
- BLOCK ERASE is initiated when the NAND Flash Controller receives the command ’60h’ from the Host.
- The Host sends the second and final command of Block Erase i.e., ’D0h’ after loading the address onto the *address register.*
- The NAND Flash Target makes the ready/busy signal go low indicating that it is done copying the address and the erase operation is under way.
- When ready/busy signal goes high, the NAND Flash Controller sends an interrupt to the Host indicating that the Block Erase process is complete and goes back to idle state.


IDLE
wr_address_from_nvm == 'hFFA
wr_data_from_nvm == 'h60
COMMAND INPUT
rg_onfi_cle <= 1
data_to_flash <= {command_register}
to target
ADDRESS INPUT 1ST CYCLE
rg_onfi_ale <= 1
buffer_address[15:0]
to target
INTERRUPT HOST
rg_interrupt <= 1
to host
ADDRESS INPUT 2ND CYCLE
buffer_address[31:16]
to target
wr_ready_busy == 0
BLOCK ERASE INITIATED
wr_ready_busy == 1
BLOCK ERASE DONE

READ STATUS

- READ STATUS is initiated when the NAND Flash Controller receives the command ’70h’ from the Host.
- After receiving the command, the NAND Flash Target sends the status of the last-selected LUN to the the NAND Flash Controller.
- This command is generally accepted by the NAND Flash Target even when it is busy.
- Status data is stored in an 8-bit *status register*.

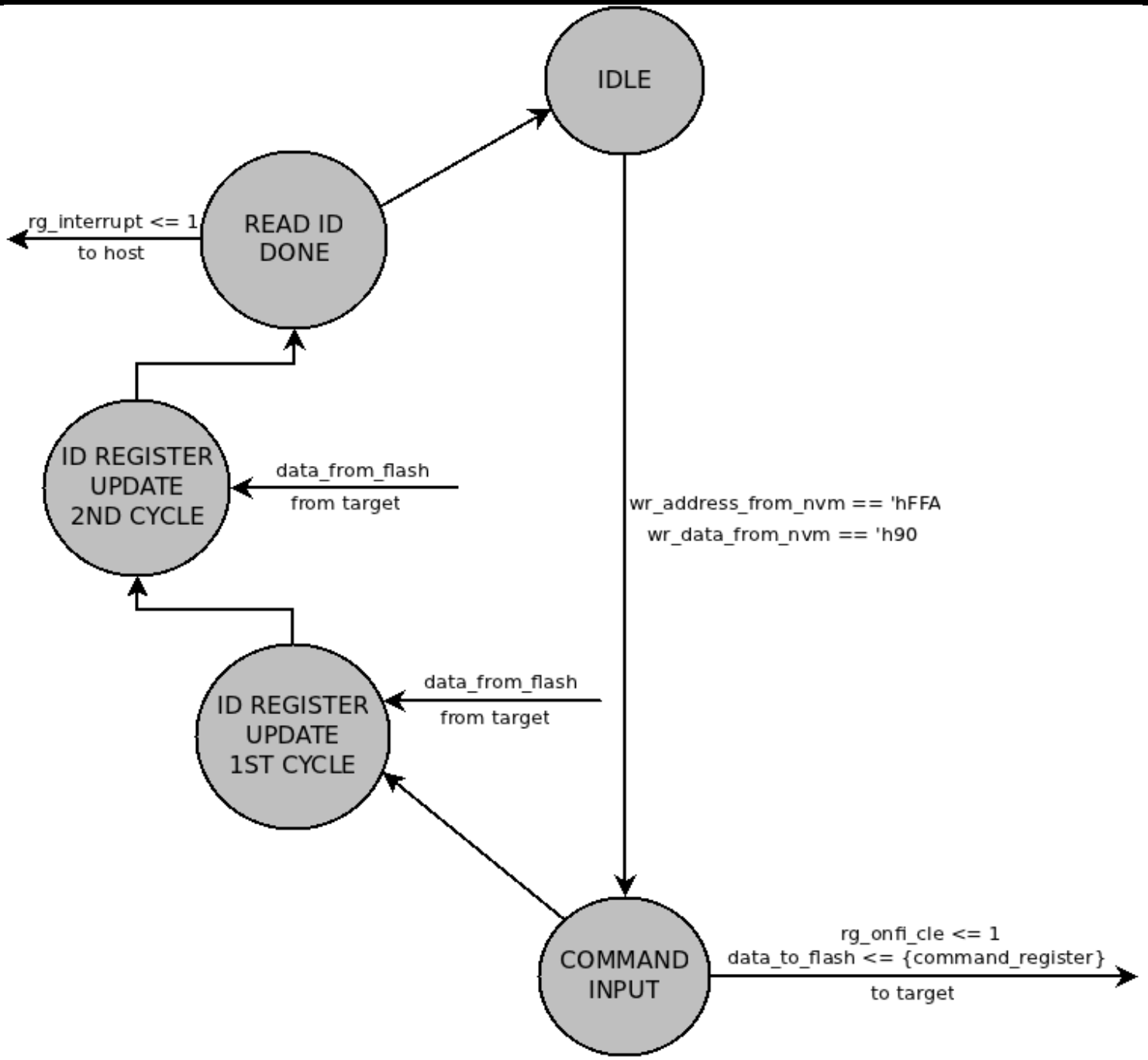
READ STATUS - STATE MACHINE



READ ID

- READ ID is initiated when the NAND Flash Controller receives the command ’90h’ from the Host.
- The READ ID (90h) command is used to read identifier codes programmed into the target.
- After receiving the command, the NAND Flash Target sends the ID data to the NAND Flash Controller.
- ID data is generally of 4-byte size.

READ ID - STATE MACHINE



READ ID RESPONSE

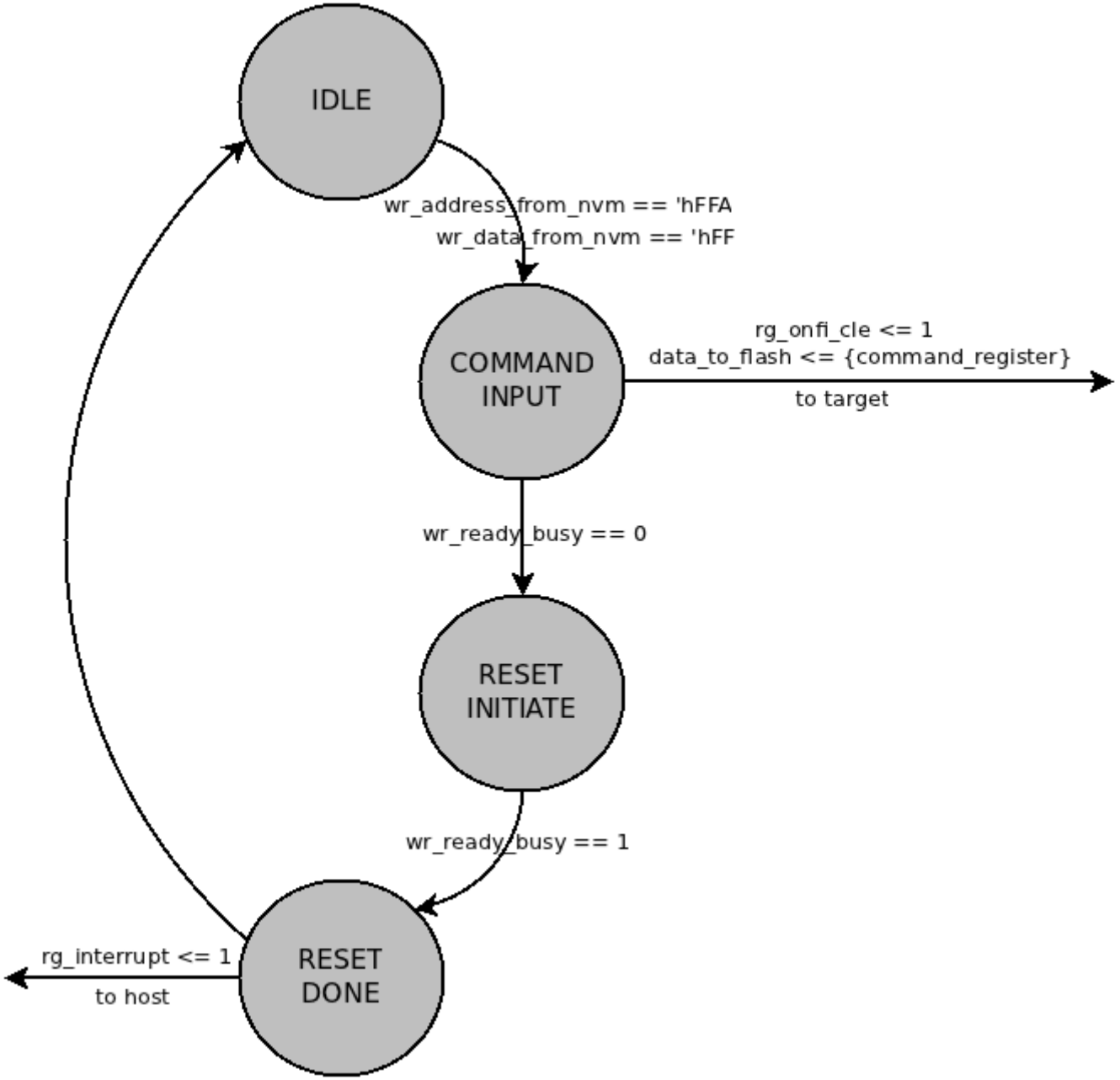
	Option	I/O7	I/O6	I/O5	I/O4	I/O3	I/O2	I/O1	I/O0	Value[1]
Byte 0 – Manufacturer ID										
Manufacturer	Micron	0	0	1	0	1	1	0	0	2Ch
Byte 1 – Device ID										
MT29F2G08AAD	2Gb, x8, 3V	1	1	0	1	1	0	1	0	DAh
MT29F2G16AAD	2Gb, x8, 3V	1	1	0	0	1	0	1	0	CAh
MT29F2G08ABD	2Gb, x8, 1.8V	1	0	1	0	1	0	1	0	AAh
MT29F2G16ABD	2Gb, x16, 1.8V	1	0	1	1	1	0	1	0	BAh
Byte 2										
Number of die per CE#	1							0	0	00b
Cell type	SLC					0	0			00b
Number of simultaneously programmed pages	1			0	0					01b
Interleaved operations between multiple die	Not supported		0							0b
Cache programming	Supported	1								1b
Byte value	MT29F2Gxxxxx	1	0	0	0	0	0	0	0	80h
Byte 3										
Page size	2KB							0	1	01b
Spare area size (bytes)	64B						1			1b
Block size (w/o spare)	128KB			0	1					01b
Organization	x8		0							0b
	x16		1							1b
Serial access (MIN)	25ns	1				0				1xxxb
Serial access (MIN)	35ns	0				0				0xxx0b
Byte value	MT29F2G08AAD	1	0	0	1	0	1	0	1	95h
	MT29F2G16AAD	1	1	0	1	0	1	0	1	D5h
Byte value	MT29F2G08ABD	0	0	0	1	0	1	0	1	15h
	MT29F2G16ABD	0	1	0	1	0	1	0	1	55h
Byte 4										
Reserved								0	0	00b
Planes per CE#	1					0	0			00b
Plane size	2Gb		1	0	1					101b
Reserved		0								0b
Byte value	MT29F2Gxx	0	1	0	1	0	0	0	0	50h

Notes: 1. b = binary; h = hexadecimal

RESET

- RESET is initiated when the NAND Flash Controller receives the command ’FFh’ from the Host.
- The NAND Flash Target makes the ready/busy signal go low indicating that the reset operation is under way.
- After the ready/busy signal goes high, the NAND Flash Controller sends an interrupt to the Host indicating that the RESET process is complete and goes back to idle state.

RESET - STATE MACHINE



TIMING CONTROL LOGIC

CE_I	ALE	CLE	WE_I	RE_I	BUS STATE
1	X	X	X	X	Standby
0	0	0	1	1	Idle
0	0	1	0	1	Command cycle
0	1	0	0	1	Address cycle
0	0	0	0	1	Data input cycle
0	0	0	1	0	Data output cycle
0	1	1	X	X	Undefined

REGISTER DESCRIPTION

Register Address	READ/WRITE	Name	Register Function	
0xFF0	Read	ID register 0	The registers the host should read following a READ ID operation.	
0xFF1	Read	ID register 1		
0xFF2	Read	ID register 2		
0xFF3	Read	ID register 3		
0xFF4	R/W	Block address [7:0]	Used to address the blocks and pages of the NAND Flash.	
0xFF5	R/W	Block address [15:8]		
0xFF6	R/W	Block address [24:16]		
0xFF7	Read	Part type and ECC	Bit [0]	0 = x8 device 1 = x16 device
			Bit [2:1]	00 = 2Gb 01 = 4Gb 10 = 8Gb
			Bit [6]	0 = No ECC 1 = ECC module connected
0xFF8	R/W	Buffer number	Bit [0]	0 = Host controls Buffer 1 1 = Host controls Buffer 2
0xFF9	Read	Status register	Used for reading the status from the NAND Flash.	
0xFFA	Write	Command register	Holds the commands the controller is to execute. 00h = PAGE READ 60h = BLOCK ERASE 70h = READ STATUS 80h = PROGRAM PAGE 90h = READ ID FFh = RESET (NAND Flash) Also holds the final commands of certain operations. 10h = PROGRAM PAGE 30h = PAGE READ D0h = BLOCK ERASE	

FILES ASSOCIATED WITH THE PROJECT

- NandFlashController.bsv
  - *1000 lines*
- NandFlashTarget.bsv
  - *700 lines*
- InterfaceNandFlashController.bsv
  - *150 lines*
- InterfaceNandFlashTarget.bsv
  - *100 lines*
- InterModuleConnection.bsv
- TestBenchNFC.bsv

DESIGN CHALLENGES

CHALLENGES FACED & DECISIONS TAKEN

CHALLENGE	IMPLEMENTATION
Minimum size of single data entity stored in buffers or BRAMs is 32-bit. The data bus NAND flash supports is 16-bit.	Synchronizing the timing of data transfer by adding wait cycles, copying the 16-bit data temporarily & sending it along with the next 16-bit data.
The target interface to be designed as ONFi interface.	The timing control logic to be designed is based upon the 5 control lines of ONFi.
The host interface to be compliant with NVM express.	Interacting with NVMe designer while designing the interface.

VERIFICATION

VERIFICATION SETUP

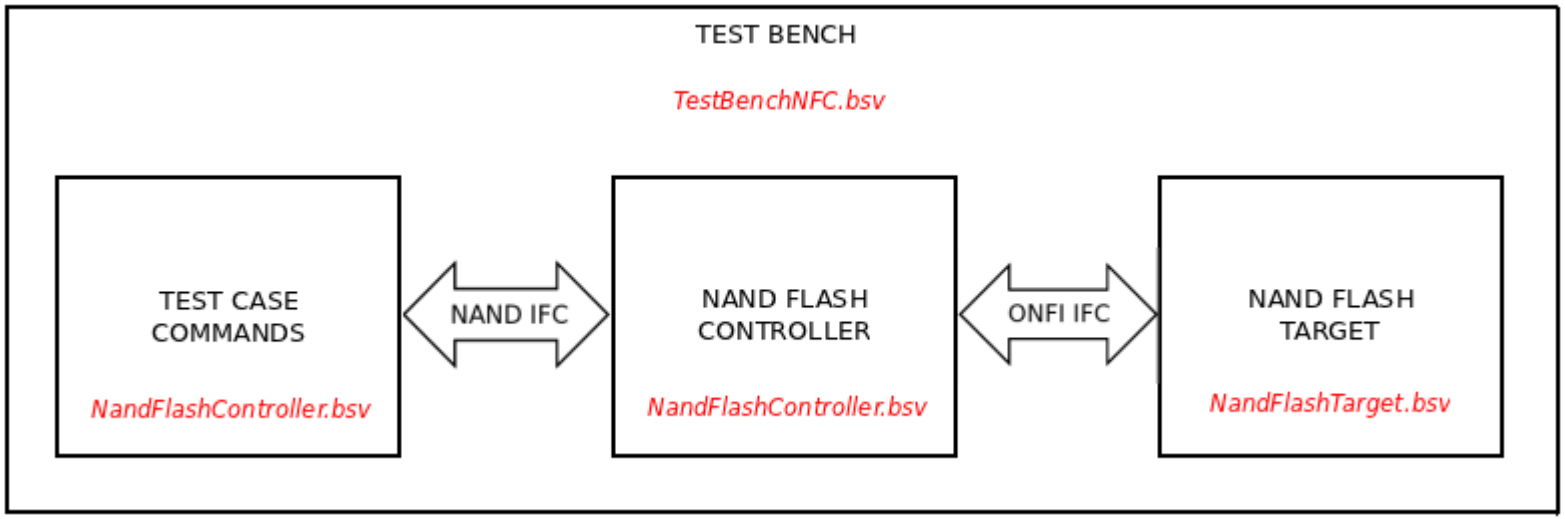


Fig : Verification setup for verifying NAND Flash Controller

VERIFICATION TEST CASES

Test cases for the following :

- Program Page
  - address & data are provided by the testbench and whether the correct data is written on the target in the right address location is verified.
- Read Page
  - address of the page to be read is given by the testbench and verified whether correct data is read.

VERIFICATION TEST CASES

- Block Erase
  - address of the block to be erased is provided by the testbench and later, checked whether the block is cleared or not.
- Read ID & Read Status
  - the specific command is given by the testbench and the data received is verified.
- Reset
  - reset command is given by the testbench to the NFC and later, verified whether the flash is reset.

VERIFICATION SETUP

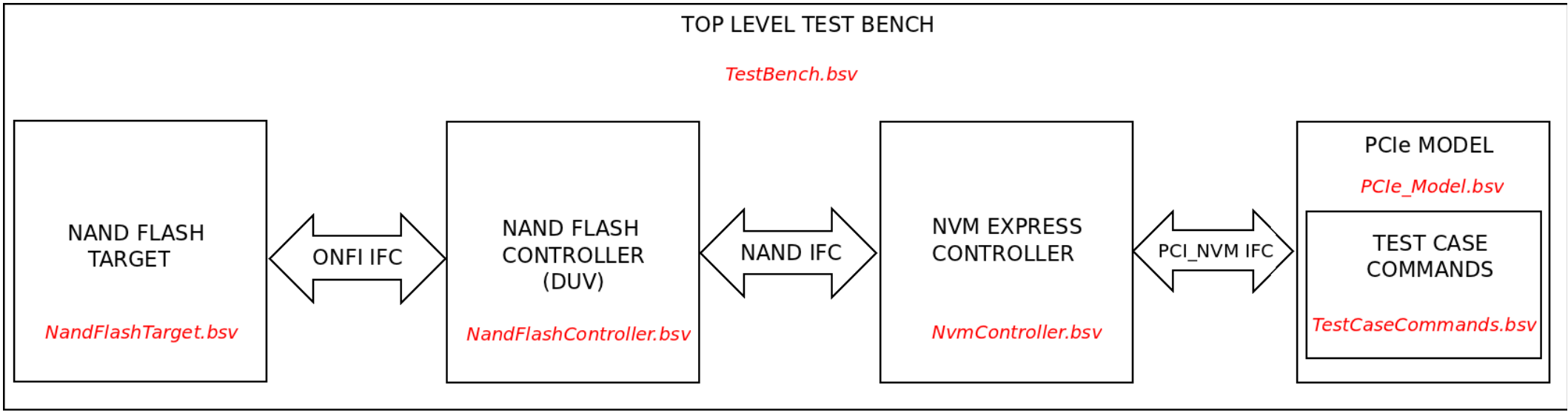


Fig : Verification setup for verifying NVMe - driven NAND Flash Controller

VERIFICATION TEST CASES

- Test cases for the following operations :
  - Read Page
  - Program Page
  - Read ID
  - Read Status
  - Reset.
- NVM express does not provide test cases for Block Erase.

SYNTHESIS REPORT

SYNTHESIS REPORT SUMMARY

FEATURE	SUMMARY
Maximum Frequency of Operation	283 MHz
No. of Slice Registers	383
No. of Slice LUTs	570
No. of LUT-FF pairs	285
No. of Block RAMs	2

CONCLUSION & FUTURE WORK

- In an industry that lacks standardization in commands, timing, architecture, and even pin-out, NAND Flash is still being rapidly adopted for solid-state drives (SSDs), and other computing and consumer applications.
- The increase in page size and write performance is a growing trend as NAND process technology shrinks, and it works very nicely with the increase in bus speed provided by ONFi improvements.
- Using CACHE MODE commands further improves the performance of NAND Flash.

REFERENCES

- Bluespec system verilog manual
- ONFi 2-3 specifications by ONFi workgroup
- ONFi PHY datasheet by ONFi workgroup
- ONFi design guide by Micron
- NAND Flash Controller datasheet by Xilinx
- NAND Flash architecture & operation by Micron
- NAND Flash Memory technical note by Micron