EEL4744

Today's Menu

- Hardware Interfacing Concepts
- Instruction Cycle, Machine Transfers, and Cycles
  - > Fetch, Decode, & Execute
  - > Memory Read & Write
- Address and Data Bus Timing
- XMEGA's EBI (**E**xternal **B**us **I**nterface)
  - > Chip Selects (CS0-CS3)

See examples on web:
`Input_Port.asm`

1

EEL4744

See doc8331, Figure 27-3

Sharing Address/Data or Address/Address Pins

- Our XMEGA (and many, but not all, other processors) have **externally accessible** address and data busses
  - > XMEGA calls this the ***EBI: External Bus Interface***
- In most situations, some of the address pins or address and data pins are **time multiplexed**, i.e., at least two distinct signals share the same pin!

2

EEL4744

Instruction Cycle, Machine Transfers, and E-Cycles

- An instruction cycle consists of a sequence of machine transfers. A machine transfer corresponds to a single machine operation (a single clock pulse) on the external buses.
- Each machine transfer is one of the following types:
  1. Opcode Fetch
  2. Memory Read
  3. Memory Write
  4. Execution - No Transfer.
- **Non-pipelined Example:** Each machine transfer requires clock cycle
  - > The corresponding time depends on the frequency of the internal clock
  - > If each the system clock is operating at 2 MHz
    - – The period is 500 ns (1 / 2 MHz)
    - – To execute a 4 cycle instruction (e.g., a STAA in a **GCPU+** with extended addressing) takes 4 × 500 ns = 2 µs

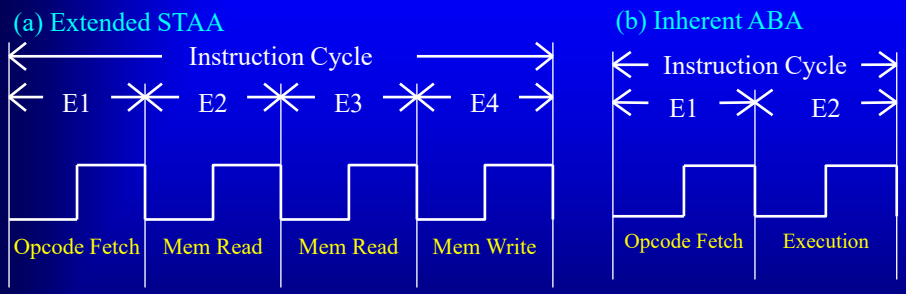
3

3

EEL4744

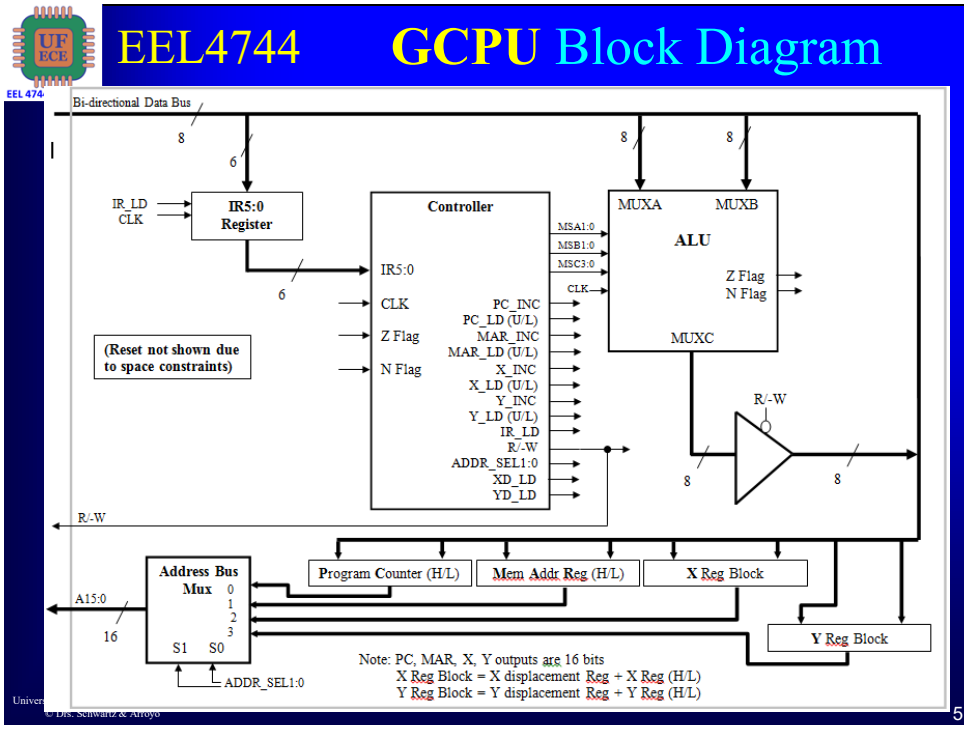
Instruction Cycles

- Instruction cycles, machine transfers, and cycles for simple instructions

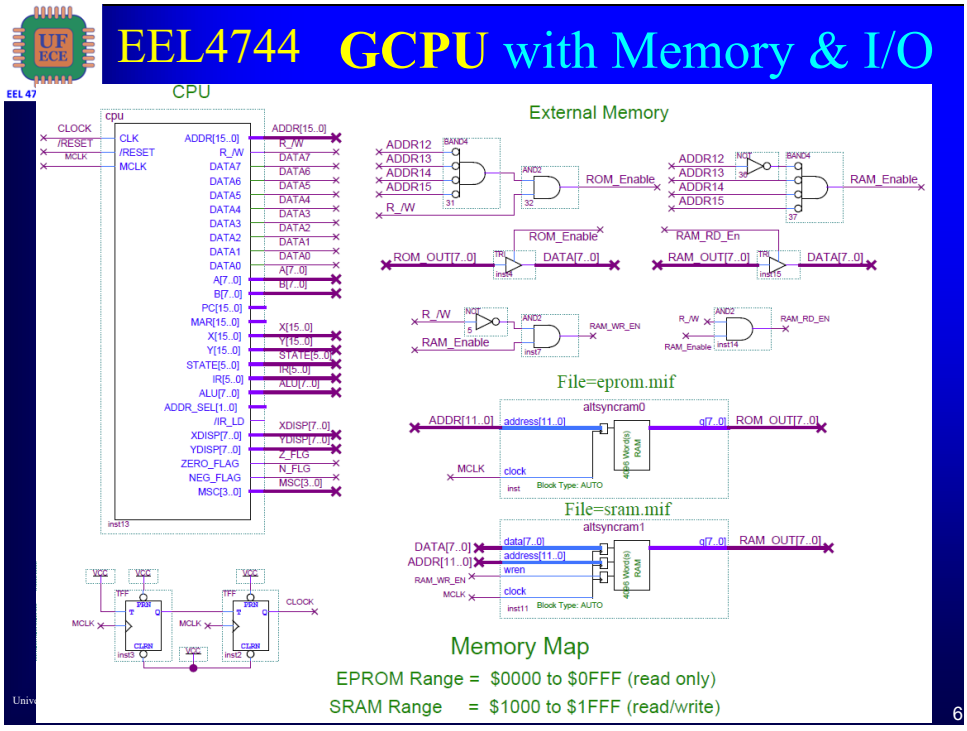


4

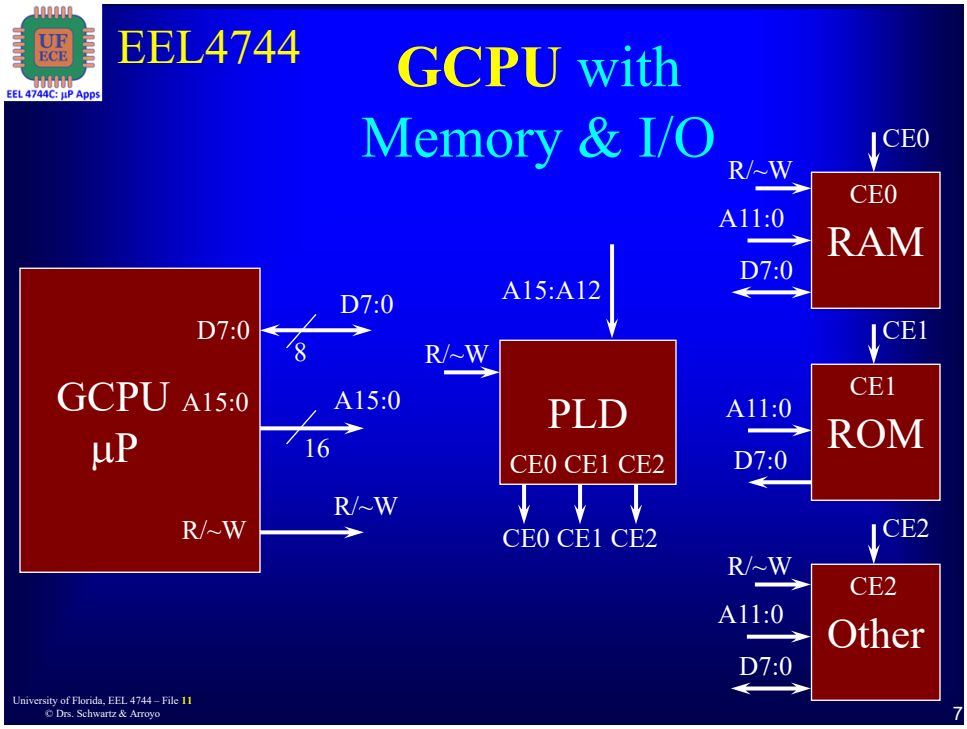
4


EEL4744 GCPU Block Diagram
Bi-directional Data Bus
IR_LD
CLK
IR5:0 Register
Controller
IR5:0
CLK
Z Flag
N Flag
(Reset not shown due to space constraints)
MSA1:0
MSB1:0
MSC3:0
CLK
PC_INC
PC_LD (U/L)
MAR_INC
MAR_LD (U/L)
X_INC
X_LD (U/L)
Y_INC
Y_LD (U/L)
IR_LD
R/-W
ADDR_SEL1:0
XD_LD
YD_LD
MUXA
MUXB
ALU
MUXC
Z Flag
N Flag
R/-W
R/-W
Address Bus Mux
A15:0
16
S1 S0
ADDR_SEL1:0
Program Counter (H/L)
Mem Addr Reg (H/L)
X Reg Block
Y Reg Block
Note: PC, MAR, X, Y outputs are 16 bits
X Reg Block = X displacement Reg + X Reg (H/L)
Y Reg Block = Y displacement Reg + Y Reg (H/L)

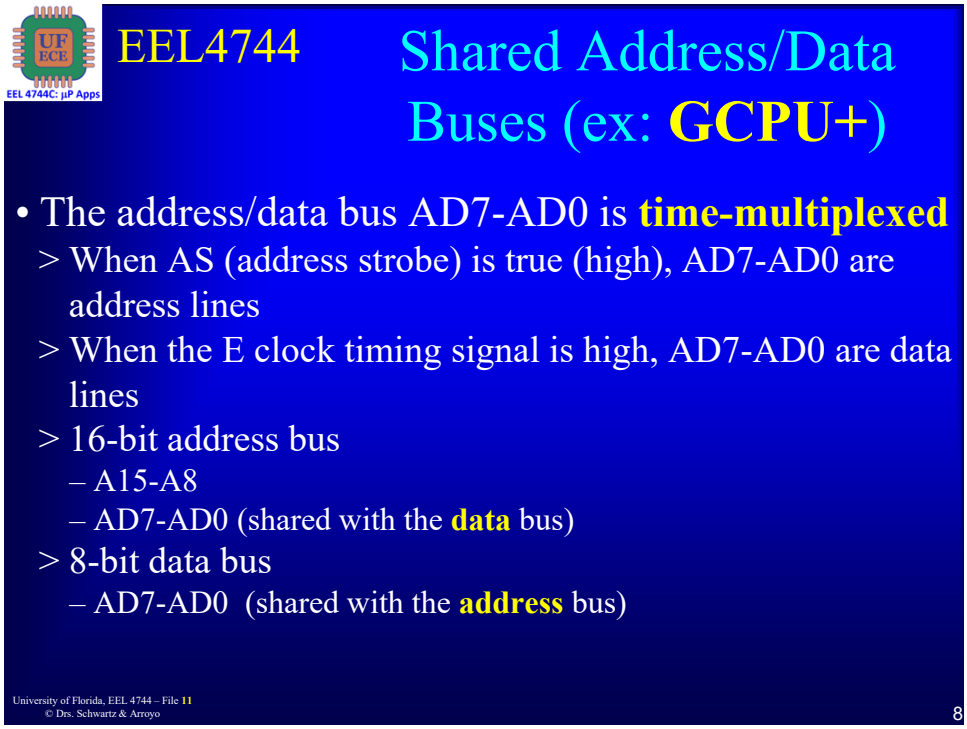
5


EEL4744 GCPU with Memory & I/O
CPU
External Memory
ROM_Enable
RAM_Enable
RAM_RD_En
RAM_WR_EN
File=eprom.mif
File=sram.mif
altsyncram0
altsyncram1
Memory Map
EPROM Range = $0000 to $0FFF (read only)
SRAM Range = $1000 to $1FFF (read/write)

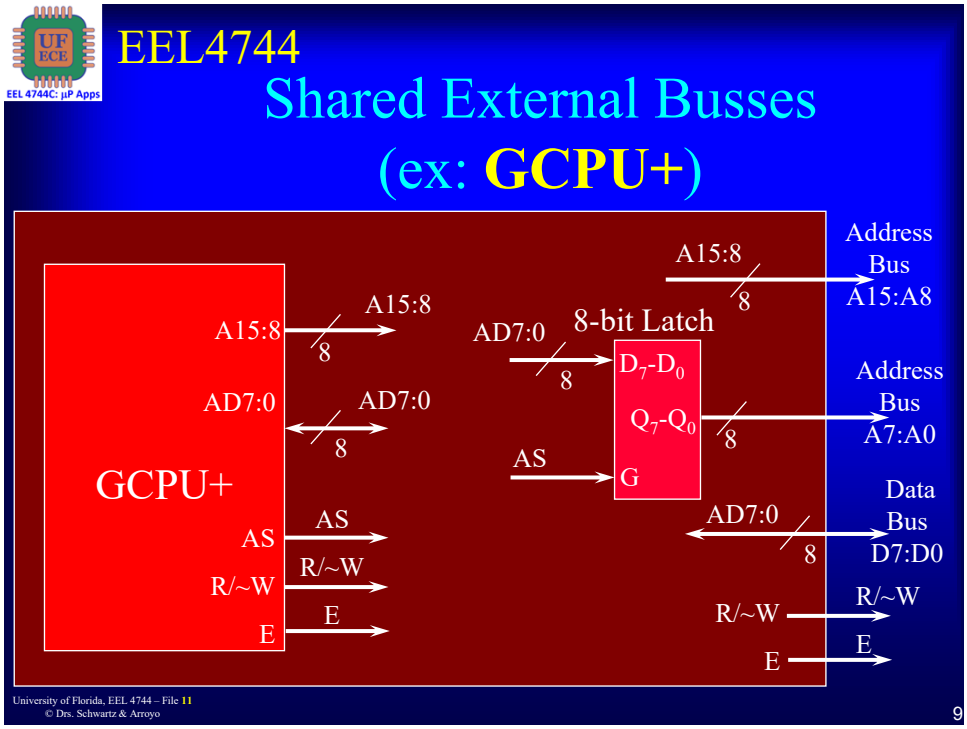
6


EEL4744
GCPU with Memory & I/O
GCPU μP
D7:0
A15:0
R/~W
8
16
PLD
CE0 CE1 CE2
A15:A12
RAM
ROM
Other
A11:0
D7:0
CE0
CE1
CE2
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
7

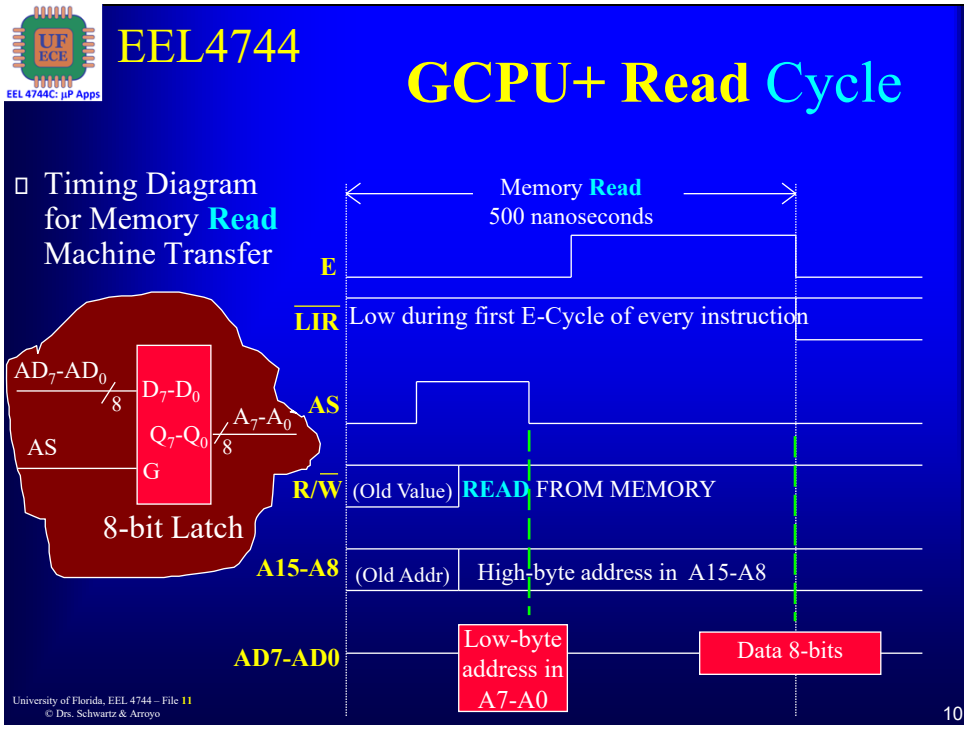
7


EEL4744
Shared Address/Data Buses (ex: GCPU+)
• The address/data bus AD7-AD0 is time-multiplexed
> When AS (address strobe) is true (high), AD7-AD0 are address lines
> When the E clock timing signal is high, AD7-AD0 are data lines
> 16-bit address bus
– A15-A8
– AD7-AD0 (shared with the data bus)
> 8-bit data bus
– AD7-AD0 (shared with the address bus)
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
8

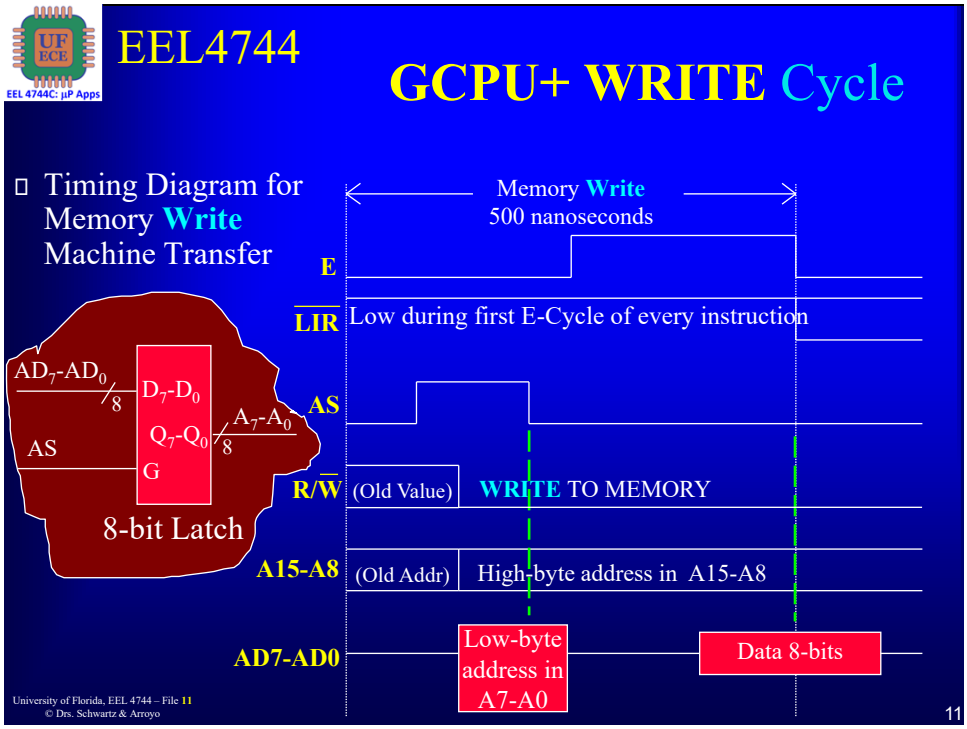
8


EEL4744
Shared External Busses
(ex: GCPU+)
GCPU+
A15:8
AD7:0
AS
R/~W
E
8-bit Latch
D7-D0
Q7-Q0
G
Address Bus A15:A8
Address Bus A7:A0
Data Bus D7:D0
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
9

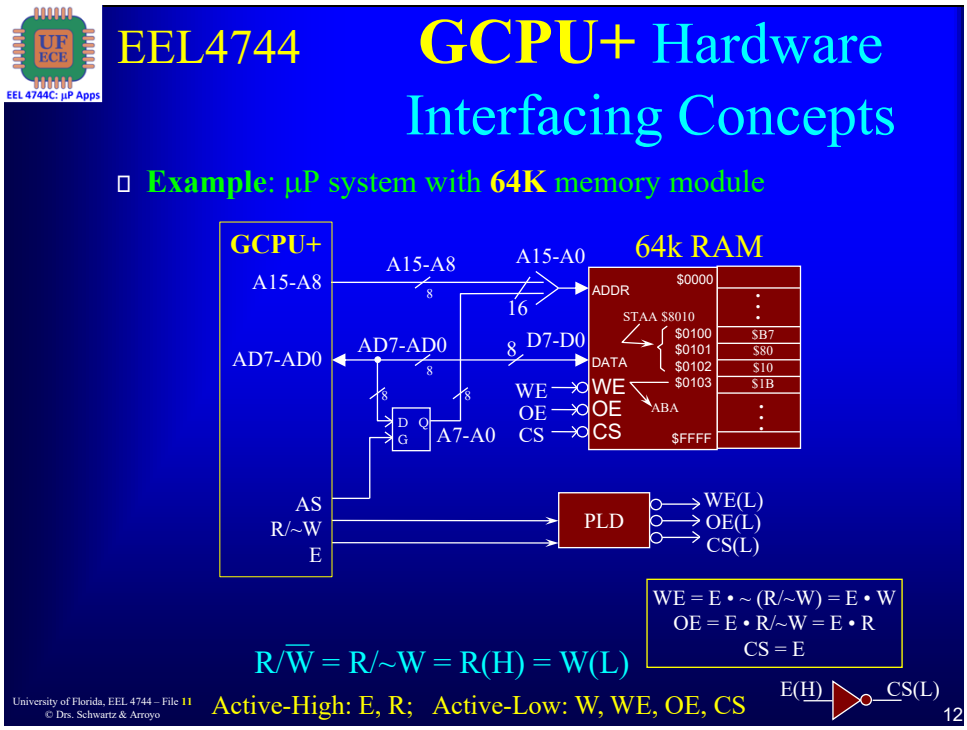
9


EEL4744
GCPU+ Read Cycle
Timing Diagram for Memory Read Machine Transfer
Memory Read
500 nanoseconds
E
LIR
Low during first E-Cycle of every instruction
AS
R/W
(Old Value)
READ FROM MEMORY
A15-A8
(Old Addr)
High-byte address in A15-A8
AD7-AD0
Low-byte address in A7-A0
Data 8-bits
AD7-AD0
D7-D0
Q7-Q0
A7-A0
G
8-bit Latch
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
10

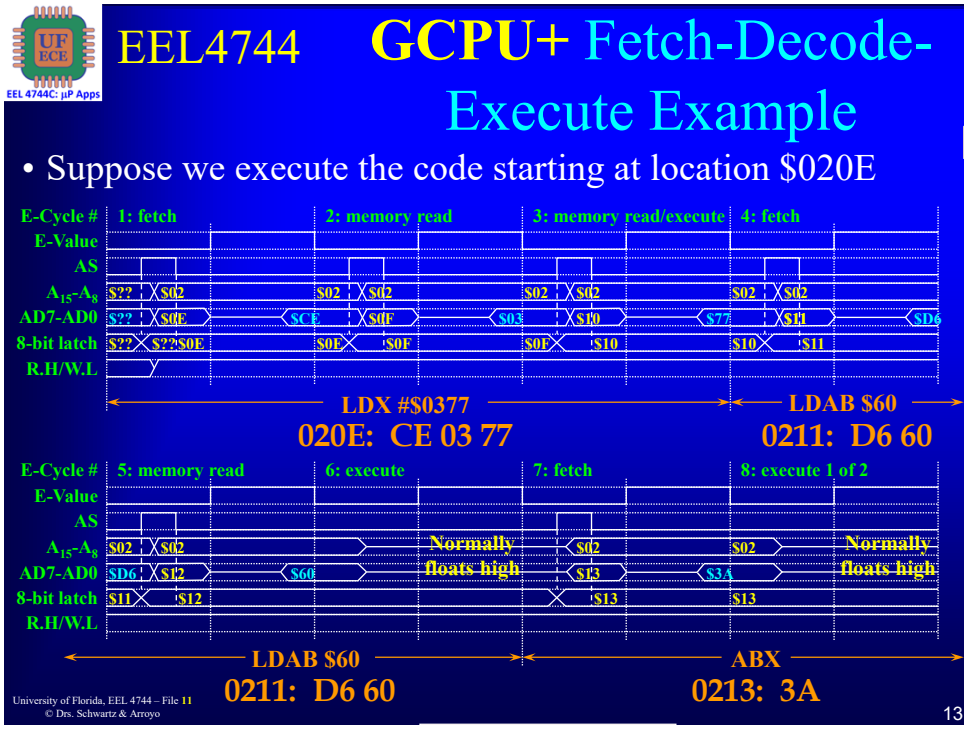
10


EEL4744
GCPU+ WRITE Cycle
Timing Diagram for Memory Write Machine Transfer
Memory Write
500 nanoseconds
E
LIR
Low during first E-Cycle of every instruction
AD7-AD0
8
D7-D0
Q7-Q0
G
AS
A7-A0
8
8-bit Latch
AS
R/W
(Old Value)
WRITE TO MEMORY
A15-A8
(Old Addr)
High-byte address in A15-A8
AD7-AD0
Low-byte address in A7-A0
Data 8-bits
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
11

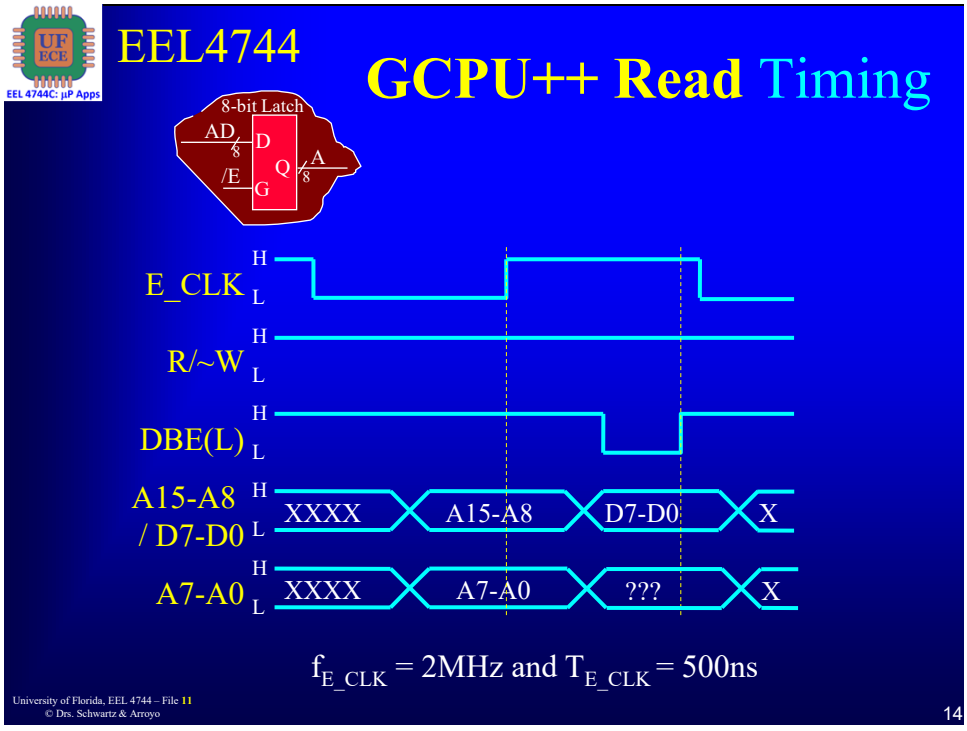
11


EEL4744
GCPU+ Hardware Interfacing Concepts
Example: µP system with 64K memory module
GCPU+
A15-A8
AD7-AD0
AS
R/~W
E
A15-A8
8
A15-A0
16
AD7-AD0
8
D7-D0
A7-A0
64k RAM
ADDR
$0000
STAA $8010
$0100
$0101
$0102
$0103
$B7
$80
$10
$1B
DATA
WE
OE
CS
ABA
$FFFF
PLD
WE(L)
OE(L)
CS(L)
WE = E • ~ (R/~W) = E • W
OE = E • R/~W = E • R
CS = E
R/W = R/~W = R(H) = W(L)
E(H)
CS(L)
Active-High: E, R; Active-Low: W, WE, OE, CS
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
12

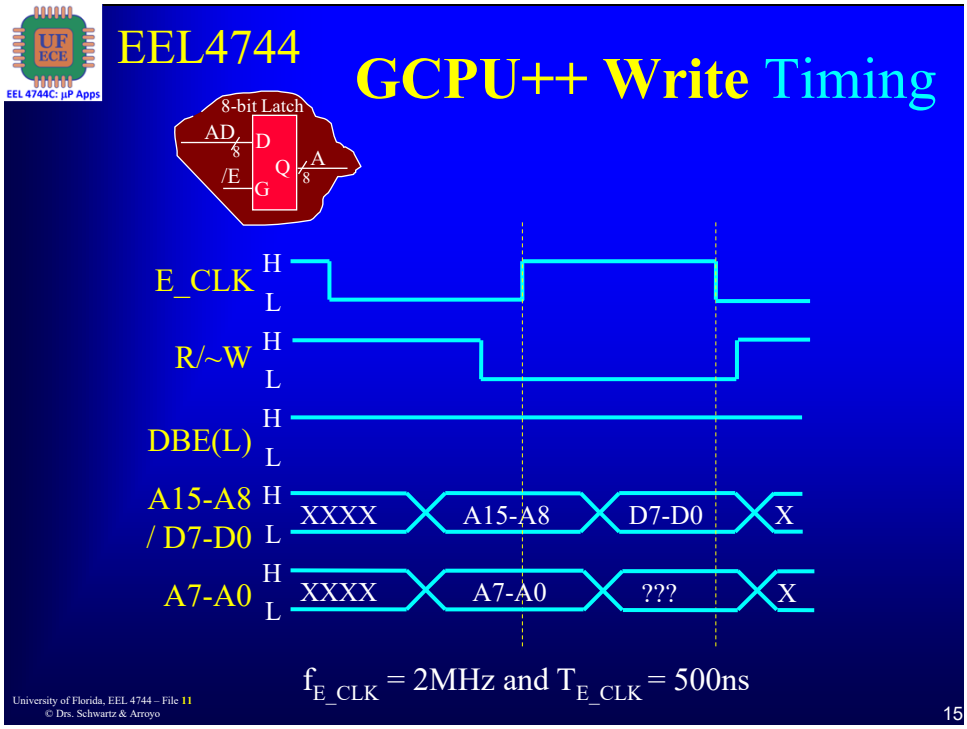
12


EEL4744
GCPU+ Fetch-Decode-Execute Example
• Suppose we execute the code starting at location $020E
E-Cycle #
E-Value
AS
A15-A8
AD7-AD0
8-bit latch
R.H/W.L
1: fetch
2: memory read
3: memory read/execute
4: fetch
LDX #$0377
020E: CE 03 77
LDAB $60
0211: D6 60
5: memory read
6: execute
7: fetch
8: execute 1 of 2
Normally floats high
LDAB $60
0211: D6 60
ABX
0213: 3A
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
13

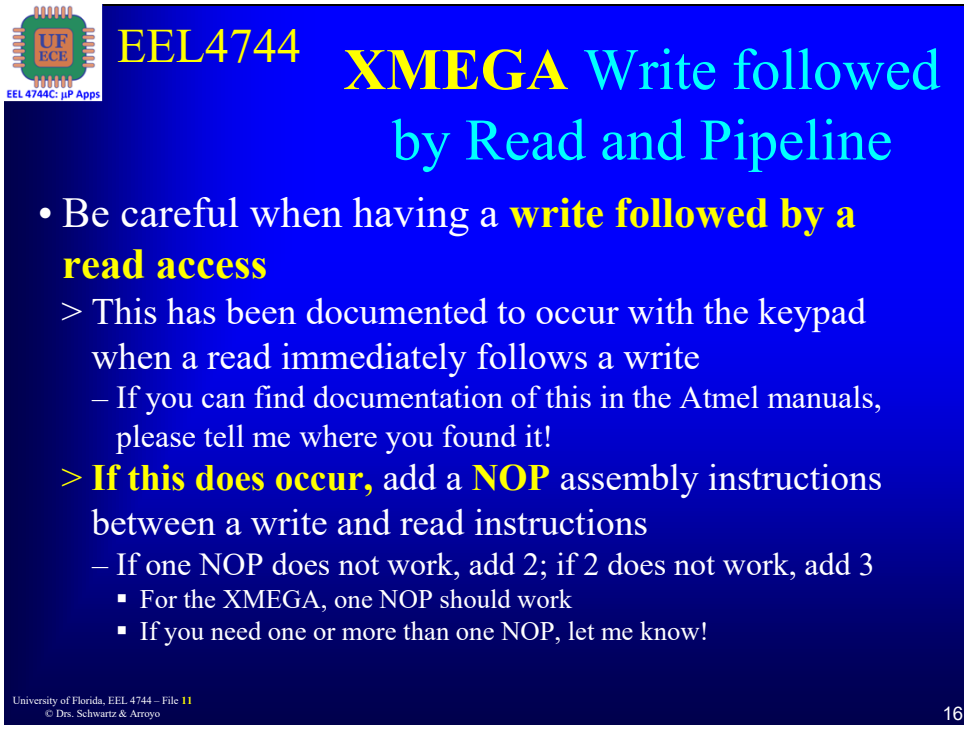
13


EEL4744
GCPU++ Read Timing
8-bit Latch
AD
D
Q
A
/E
G
E_CLK
R/~W
DBE(L)
A15-A8 / D7-D0
A7-A0
XXXX
A15-A8
D7-D0
X
XXXX
A7-A0
???
X
f_{E_CLK} = 2MHz and T_{E_CLK} = 500ns
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
14

14


EEL4744
GCPU++ Write Timing
8-bit Latch
AD
8
D
Q
A
8
/E
G
E_CLK
R/~W
DBE(L)
A15-A8
/ D7-D0
A7-A0
H
L
XXXX
A15-A8
D7-D0
X
XXXX
A7-A0
???
X
f_{E_CLK} = 2MHz and T_{E_CLK} = 500ns
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
15

15


EEL4744
XMEGA Write followed by Read and Pipeline
• Be careful when having a write followed by a read access
> This has been documented to occur with the keypad when a read immediately follows a write
– If you can find documentation of this in the Atmel manuals, please tell me where you found it!
> If this does occur, add a NOP assembly instructions between a write and read instructions
– If one NOP does not work, add 2; if 2 does not work, add 3
▪ For the XMEGA, one NOP should work
▪ If you need one or more than one NOP, let me know!
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
16

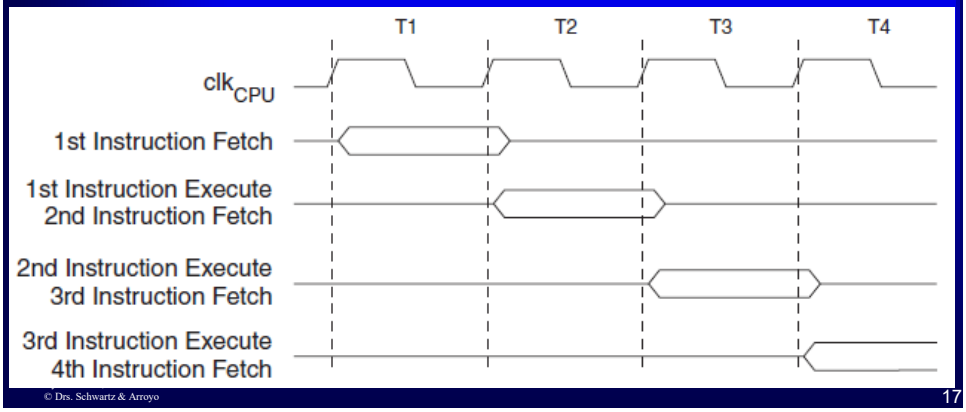
16

EEL4744

XMEGA Pipeline: Execution Timing

- The parallel instruction fetches and instruction executions timing

See doc8331, Figure 3-2



 17

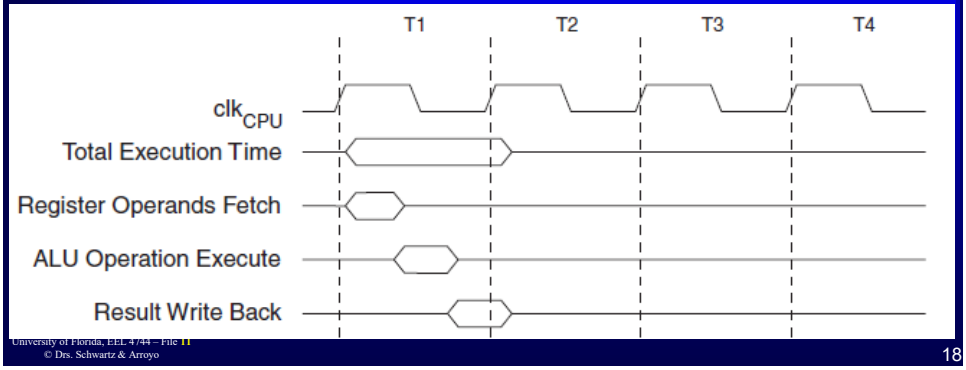
17

EEL4744

XMEGA Pipeline: Execution Timing

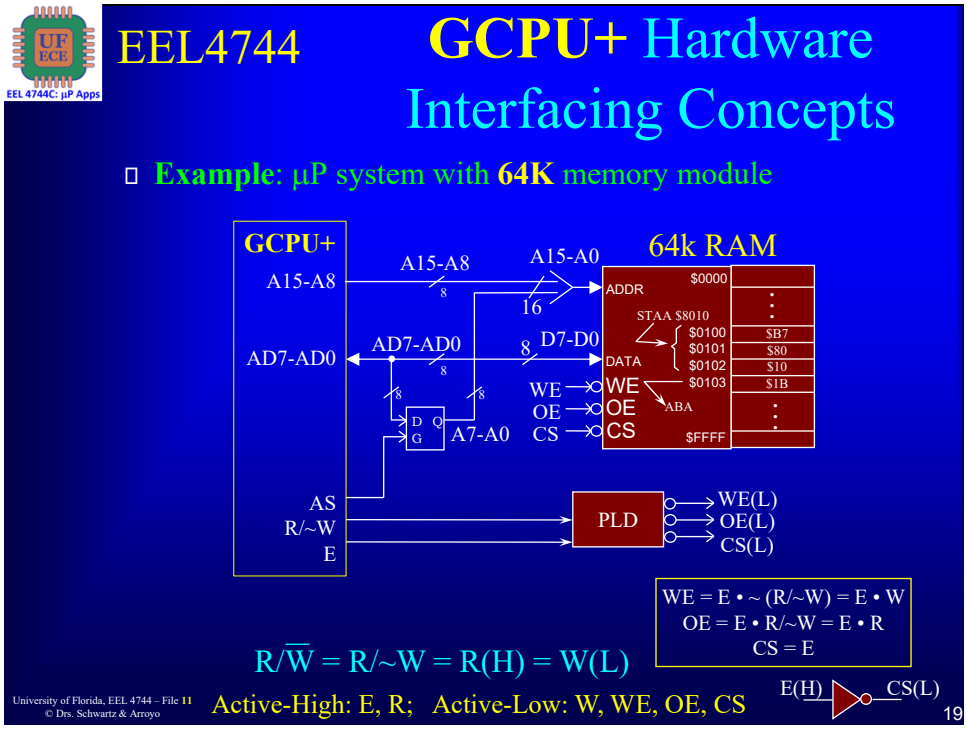
See doc8331, Figure 3-3

- Single cycle ALU instruction
  - > In a single clock cycle, an ALU operation using two register operands is executed and the result is stored back to the destination register

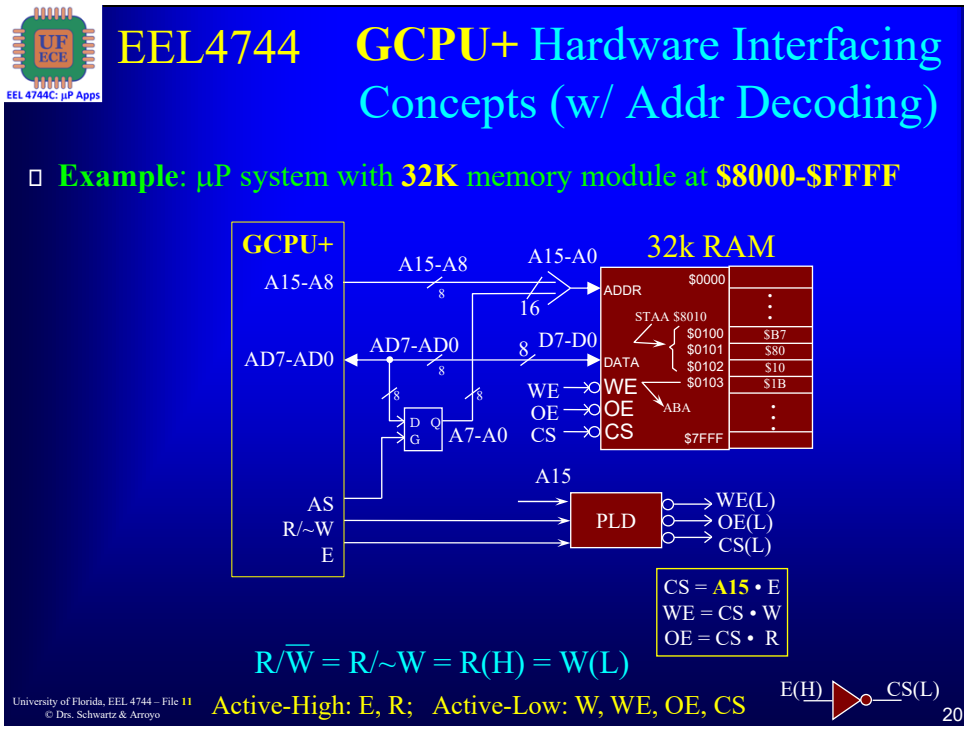


 18

18


EEL4744
GCPU+ Hardware Interfacing Concepts
Example: µP system with 64K memory module
GCPU+
A15-A8
AD7-AD0
AS
R/~W
E
A15-A8
A15-A0
16
AD7-AD0
8
D7-D0
D Q
G
A7-A0
WE
OE
CS
64k RAM
ADDR
DATA
$0000
STAA $8010
$0100
$0101
$0102
$0103
$B7
$80
$10
$1B
ABA
$FFFF
PLD
WE(L)
OE(L)
CS(L)
WE = E • ~ (R/~W) = E • W
OE = E • R/~W = E • R
CS = E
R/W = R/~W = R(H) = W(L)
Active-High: E, R; Active-Low: W, WE, OE, CS
E(H)
CS(L)
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
19

19

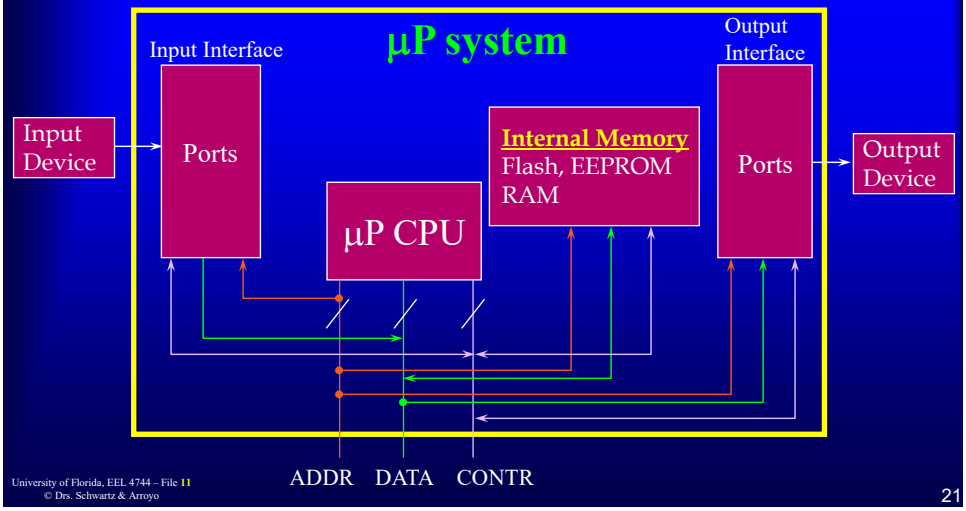

EEL4744
GCPU+ Hardware Interfacing Concepts (w/ Addr Decoding)
Example: µP system with 32K memory module at $8000-$FFFF
GCPU+
A15-A8
AD7-AD0
AS
R/~W
E
A15-A8
A15-A0
16
AD7-AD0
8
D7-D0
D Q
G
A7-A0
WE
OE
CS
32k RAM
ADDR
DATA
$0000
STAA $8010
$0100
$0101
$0102
$0103
$B7
$80
$10
$1B
ABA
$7FFF
A15
PLD
WE(L)
OE(L)
CS(L)
CS = A15 • E
WE = CS • W
OE = CS • R
R/W = R/~W = R(H) = W(L)
Active-High: E, R; Active-Low: W, WE, OE, CS
E(H)
CS(L)
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
20

20

EEL4744

Hardware/Interfacing Review

- Block Diagram of a Basic Microprocessor System

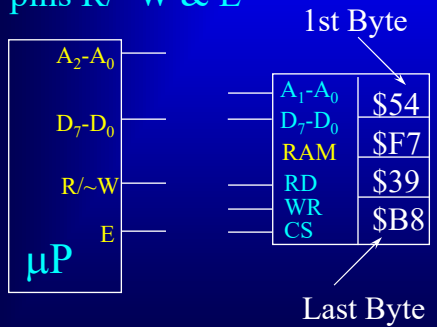


21

EEL4744

GCPU- Hardware/Interfacing

- Example R1: Add a 4×8 bit (4-bytes) RAM module to a hypothetical µP with 3 address pins, 8 data pins and control pins R/~W & E



- D_7-D_0 on the µP connect to D_7-D_0 on the RAM
- RD = E • R/~W
- WR = E • (R/~W)'
- Two of the three address lines go to A1-A0 on the RAM; CS = A_i

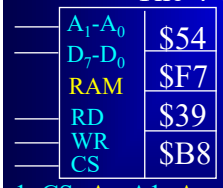
1. CS=A_2; A1=A_1; A0=A_0
2. CS=/A_2; A1=A_1; A0=A_0
3. CS=A_1; A1=A_2; A0=A_0
4. CS=A_0; A1=A_2; A0=A_1

22

EEL4744

GCPU-Hardware/Interfacing

- What are the consequences of these choices?
  - > Choice 1: **CS=A_2; A1=A_1; A0=A_0**
    - – When the µP issues address 000; the RAM does not respond since **CS=A_2**=0; similarly for addresses 001, 010, 011
    - – For address 100 the µP reads $54, for 101 the µP reads $F7, for 110 the µP reads $39, for 111 the µP reads $B8
    - – The 4-byte RAM starts at address 100



1. CS=A_2; A1=A_1; A0=A_0
2. CS=/A_2; A1=A_1; A0=A_0
3. CS=A_1; A1=A_2; A0=A_0
4. CS=A_0; A1=A_2; A0=A_1

Addr	Choice 1	Choice 2	Choice 3	Choice 4
000	**None**	$54	None	None
001	**None**	$F7	None	$54
010	**None**	$39	$54	None
011	**None**	$B8	$F7	$F7
100	**$54**	None	None	None
101	**$F7**	None	None	$39
110	**$39**	None	$39	None
111	**$B8**	None	$B8	$B8

University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
23

23

EEL4744

Conclusions from Example R1

- The data in the RAM will not be accessed contiguously unless we connect the matching contiguous low order lines to the RAM, i.e., A1=A_1 and A0=A_0
- We have a choice of CS=/A_2 or CS=A_2
  - > If want the RAM in the "**low memory range,**" choose **CS=** /A_2
  - > If want the RAM in the "**high memory range,**" choose **CS** = A_2
- For **contiguous access** we always connect the **low order** address pins to **all** the RAM address pins
- CS = *f* (**unused high order** address lines). If we have **m** unused address lines we will have **2^m** possible starting addresses for the contiguous memory block

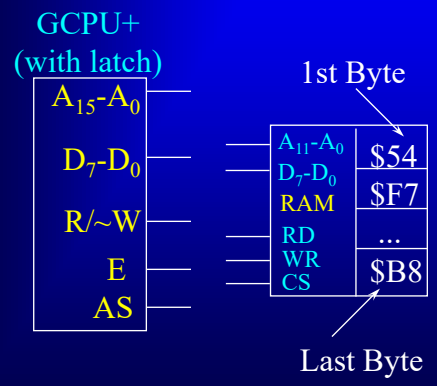
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
24

24

EEL4744

GCPU+ Hardware/Interfacing

- Example R2: Add a 4k×8 bit (4k-bytes, 4K) RAM module to the GCPU+



- D_7-D_0 on the µP connect to D_7-D_0 on the RAM
- RD = E • R/~W
- WR = E • (R/~W)'
- µP A_{11}-A_0 to A_{11}-A_0 on the RAM; CS=$f(A_{15}$-$A_{12})$

1. CS=/A_{15}•/A_{14}•/A_{13}•/A_{12}•E
2. CS=/A_{15}•/A_{14}•/A_{13}•A_{12}•E
3. CS=/A_{15}•/A_{14}•A_{13}•/A_{12}•E
4. CS=/A_{15}•/A_{14}•A_{13}•A_{12}•E

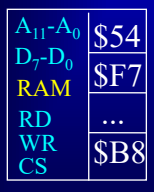
25

EEL4744

GCPU+ Hardware/Interfacing Review

- What are the consequences of these choices?
  - > Choice 1: CS=/A_{15}•/A_{14}•/A_{13}•/A_{12}•E
    - – When the µP issues address 0000 xxxx xxxx xxxx; the RAM responds since CS=1•1•1•1 (address lines are active-high)
    - – For address 0000 0000 0000 0000 the µP reads $54, for $0001 the µP reads $F7, for $0FFF the µP reads $B8
    - – The 4K-byte RAM starts at address $0000

1. CS=/A_{15}•/A_{14}•/A_{13}•/A_{12}•E
2. CS=/A_{15}•/A_{14}•/A_{13}•A_{12}•E
3. CS=/A_{15}•/A_{14}•A_{13}•/A_{12}•E
4. CS=/A_{15}•/A_{14}•A_{13}•A_{12}•E



Address	C1	C2	C3	C4
0000 0000 0000 0000	**$54**	None	None	None
0000 0000 0000 0001	**$F7**	None	None	None
0001 0000 0000 0000	**None**	$54	None	None
0001 0000 0000 0001	**None**	$F7	None	None
0010 0000 0000 0000	**None**	None	$54	None
0010 0000 0000 0001	**None**	None	$F7	None
0011 0000 0000 0000	**None**	None	None	$54
0011 0000 0000 0001	**None**	None	None	$F7

26

EEL4744

GCPU+ Conclusions from Example R2

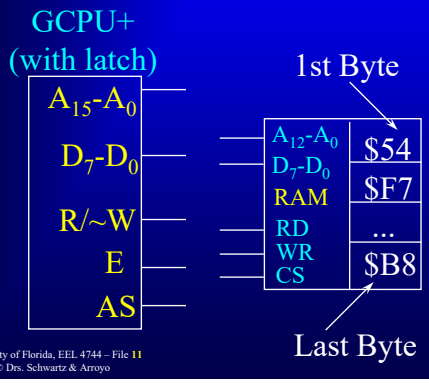
- For choice 1, CS=/A_{15}•/A_{14}•/A_{13}•/A_{12}•E the 4K memory block begins at address \$0000; the range is \$0000-\$0FFF
- For choice 2, CS=/A_{15}•/A_{14}•/A_{13}•A_{12}•E the 4K memory block begins at address \$1000; the range is \$1000-\$1FFF
- For choice 3, CS=/A_{15}•/A_{14}•A_{13}•/A_{12}•E the 4K memory block begins at address \$2000; the range is \$2000-\$2FFF
- For choice 4, CS=/A_{15}•/A_{14}•A_{13}•A_{12}•E the 4K memory block begins at address \$3000; the range is \$3000-\$3FFF
- There are **2^4=16** choices for a starting address for the 4K block, mainly, \$0000, \$1000, \$2000,..., \$E000, \$F000

27

EEL4744

GCPU+ Hardware/Interfacing

- Example R3: Add a 8k×8 bit (8k-bytes, 8K) RAM module to the GCPU+ starting at address \$4000



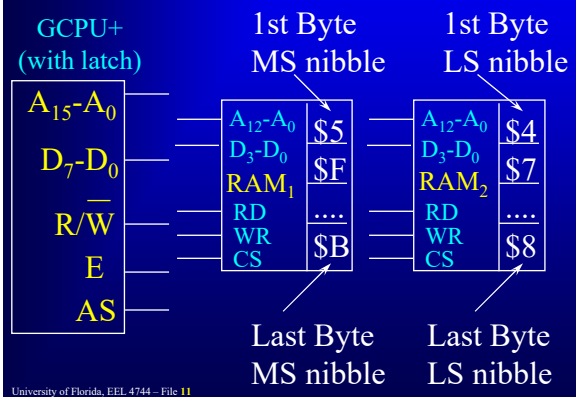
- D_7-D_0 on the µP connect to D_7-D_0 on the RAM
- RD = E • R/~W
- WR = E • (R/~W)'
- µP A_{12}-A_0 to A_{12}-A_0 on the RAM; CS=*f*(A_{15}-A_{13})
- Since starting address is **010**0 0000 0000 0000 the only choice is CS=/A_{15}•A_{14}•/A_{13}•E
- The address range is \$4000-\$5FFF

28

EEL4744

GCPU+ Hardware/Interfacing

- Example R4: Add a 8K of RAM using two 8k x 4 RAM chips to the GCPU+ starting at address $A000



- D_3-D_0 on the µP to D_3-D_0 on RAM_2 (LS nibble)
- D_7-D_4 on the µP to D_3-D_0 on RAM_1 (MS nibble)
- WR = E • (R/~W)'
- RD = E • R/~W
- µP A_{12}-A_0 to A_{12}-A_0 on the RAM; CS=$f(A_{15}$-$A_{13})$
- For starting address **1010** 0000 0000 0000 the only choice is CS=A_{15}•/A_{14}•A_{13}•E
- The address range is $A000-$BFFF

29

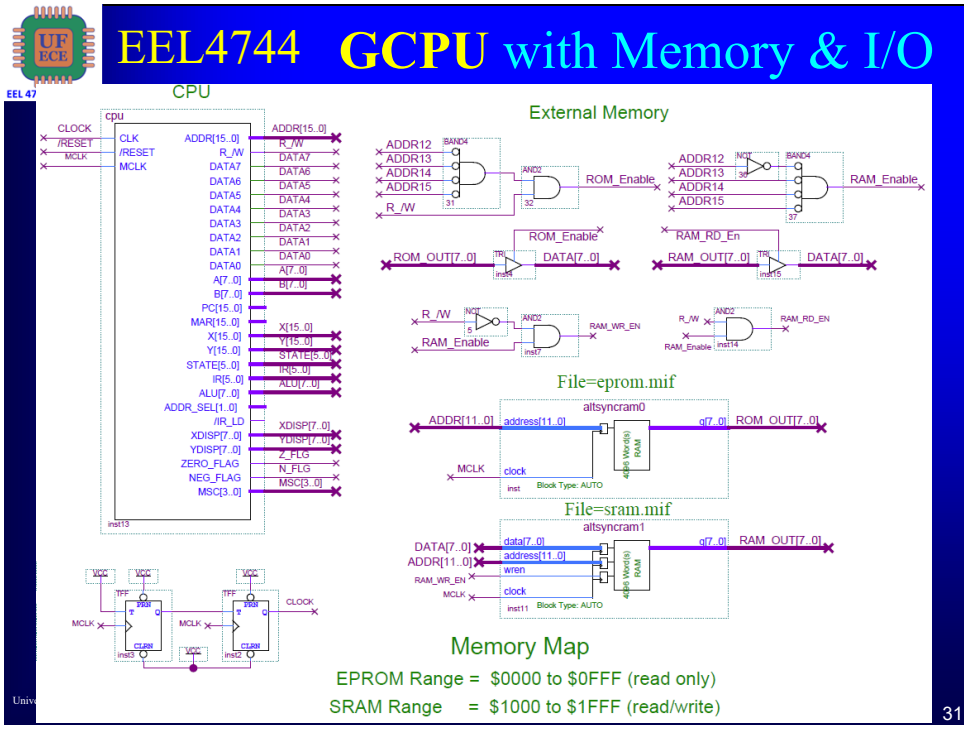
EEL4744

XMEGA Data Memory Map

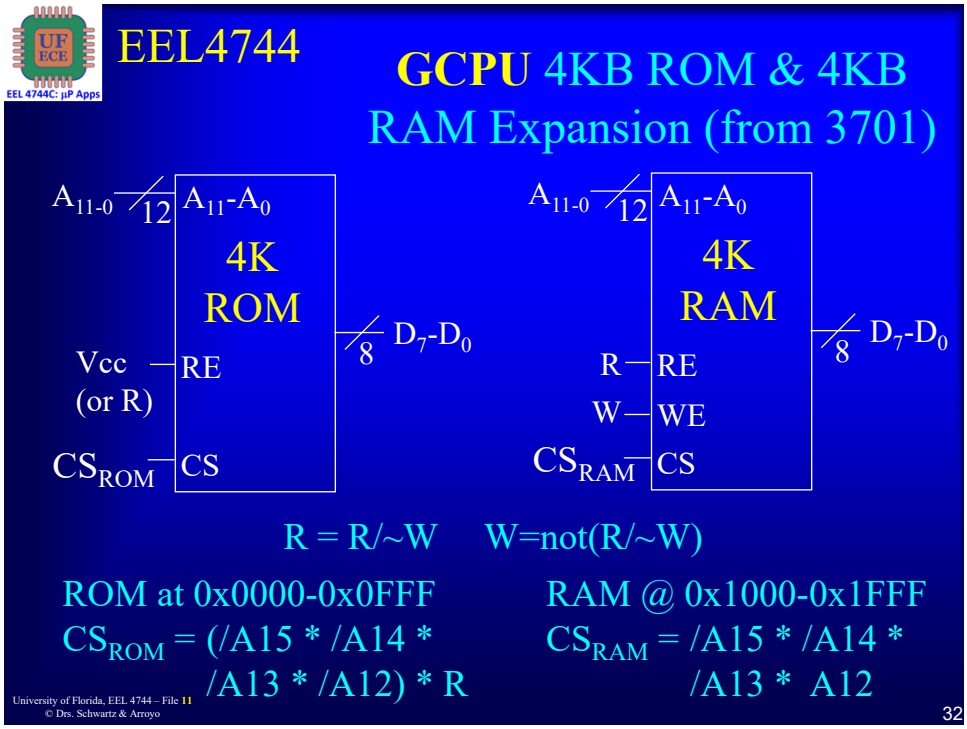
See doc8385, Figure 7-2 CORRECTED

Addr (hex)	Description
0 – 0FFF	I/O Registers (4kB) [0 – 0xBC3 on ours] (Configuration registers)
1000 – 17FF	EEPROM (2kB)
1800 – 1FFF	Reserved
2000 – 3FFF	Internal SRAM (8kB)
4000 – FF FFFF	**External "Memory" (~16MB)**

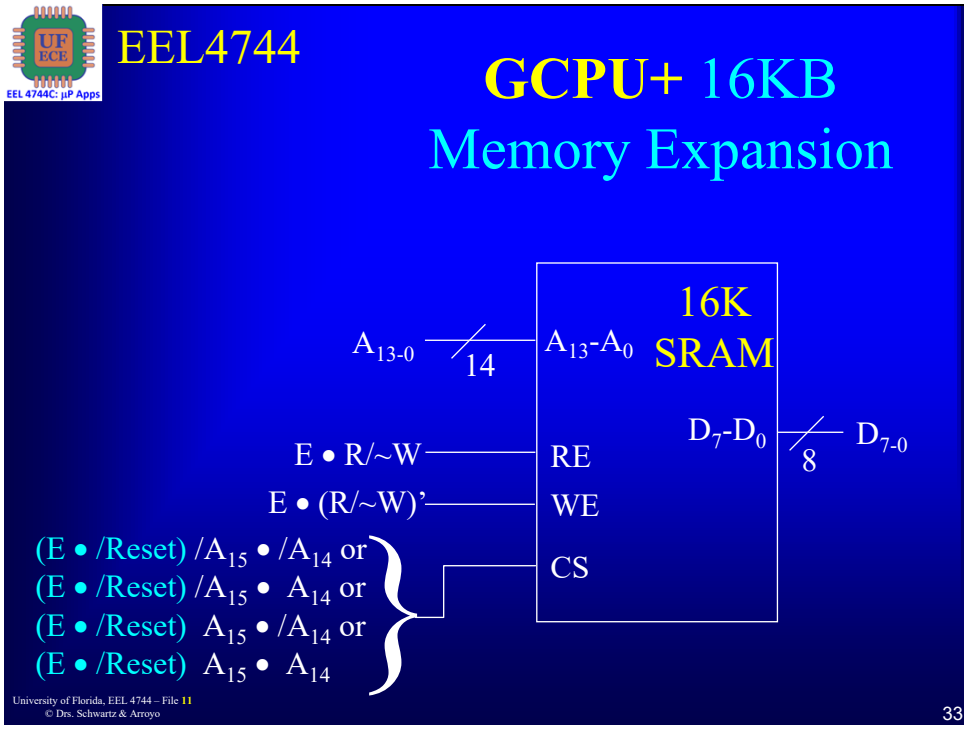
30


EEL4744 GCPU with Memory & I/O
CPU
External Memory
ROM_Enable
RAM_Enable
File=eprom.mif
File=sram.mif
Memory Map
EPROM Range = $0000 to $0FFF (read only)
SRAM Range = $1000 to $1FFF (read/write)
31

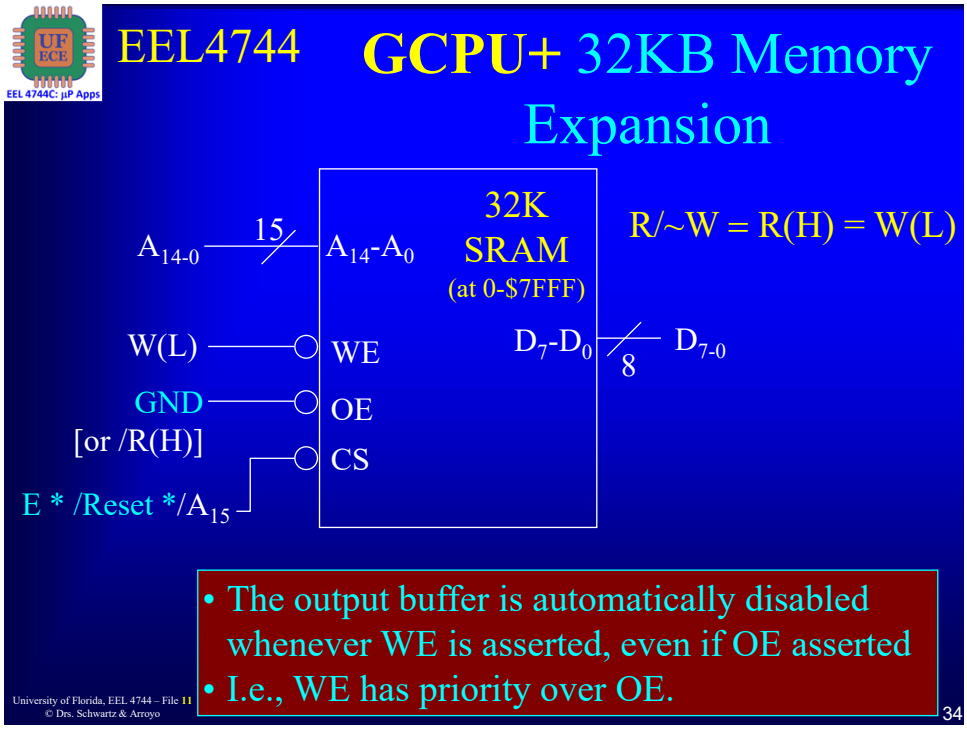
31


EEL4744
GCPU 4KB ROM & 4KB RAM Expansion (from 3701)
A11-0
12
A11-A0
4K ROM
Vcc (or R)
RE
CSROM
CS
8
D7-D0
4K RAM
R
W
WE
CSRAM
R = R/~W
W=not(R/~W)
ROM at 0x0000-0x0FFF
CSROM = (/A15 * /A14 * /A13 * /A12) * R
RAM @ 0x1000-0x1FFF
CSRAM = /A15 * /A14 * /A13 * A12
32

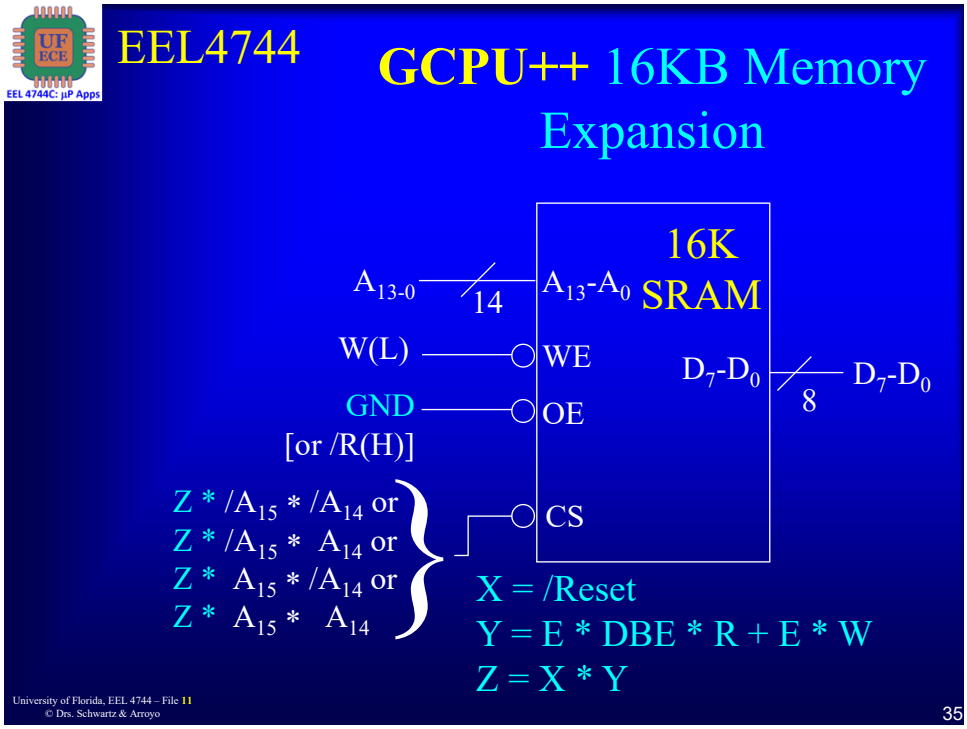
32


EEL4744
GCPU+ 16KB Memory Expansion
16K SRAM
A13-0
14
A13-A0
D7-D0
8
D7-0
E • R/~W
RE
E • (R/~W)'
WE
(E • /Reset) /A15 • /A14 or
(E • /Reset) /A15 • A14 or
(E • /Reset) A15 • /A14 or
(E • /Reset) A15 • A14
CS
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
33

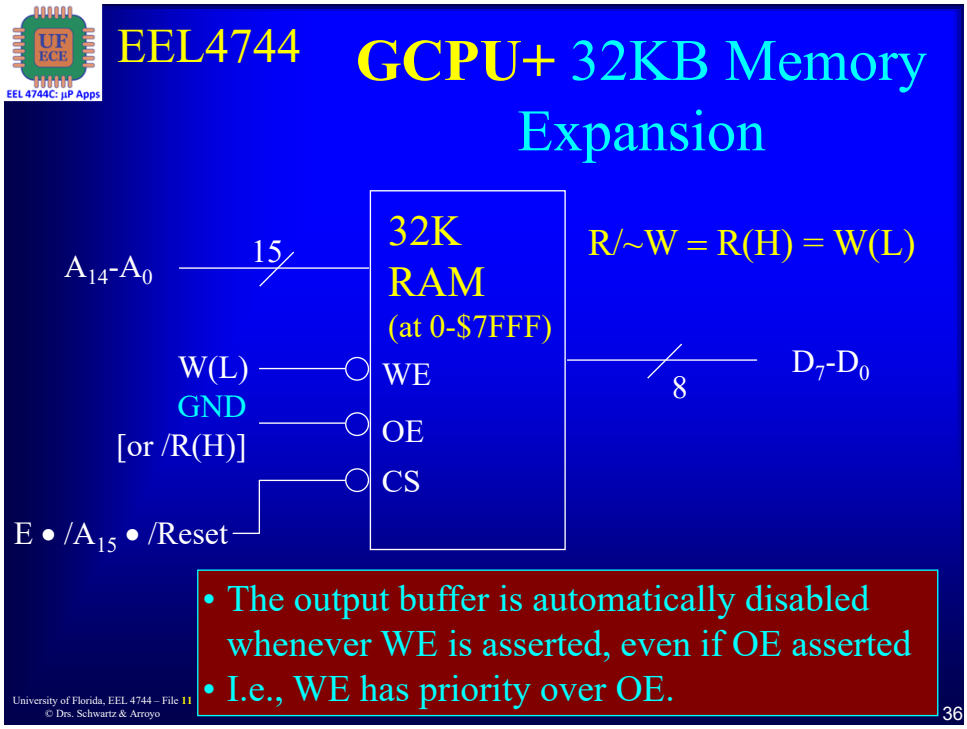
33


EEL4744
GCPU+ 32KB Memory Expansion
32K SRAM
(at 0-$7FFF)
A14-0
15
A14-A0
R/~W = R(H) = W(L)
D7-D0
8
D7-0
W(L)
WE
GND
[or /R(H)]
OE
CS
E * /Reset */A15
• The output buffer is automatically disabled whenever WE is asserted, even if OE asserted
• I.e., WE has priority over OE.
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
34

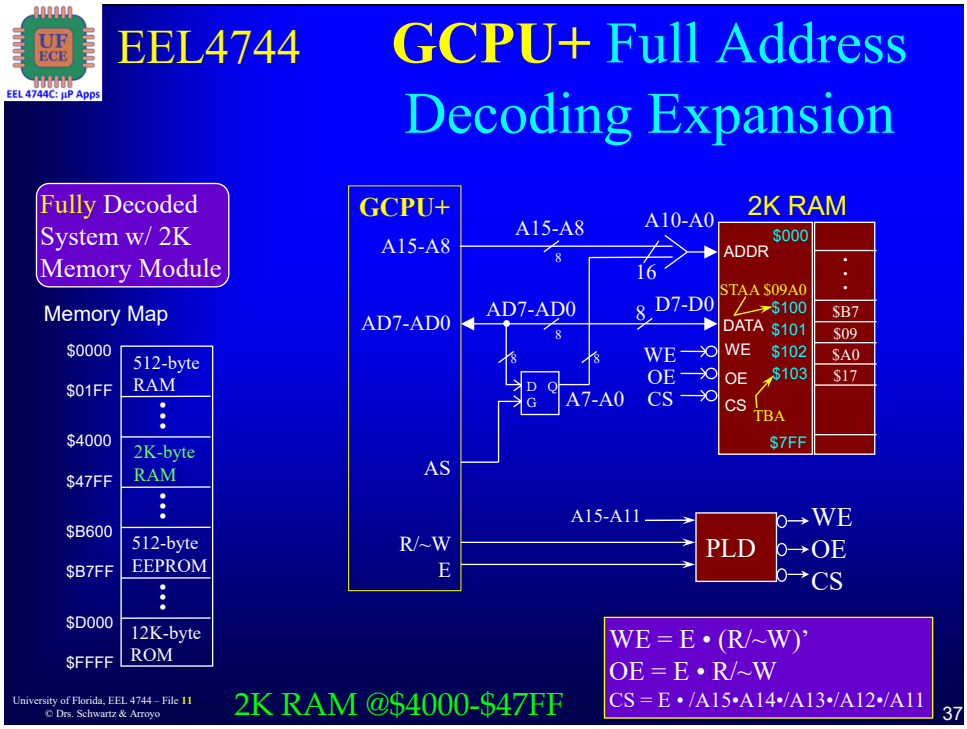
34


EEL4744
GCPU++ 16KB Memory Expansion
16K SRAM
A13-0
14
A13-A0
W(L)
WE
GND
[or /R(H)]
OE
D7-D0
8
D7-D0
Z * /A15 * /A14 or
Z * /A15 * A14 or
Z * A15 * /A14 or
Z * A15 * A14
CS
X = /Reset
Y = E * DBE * R + E * W
Z = X * Y
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
35

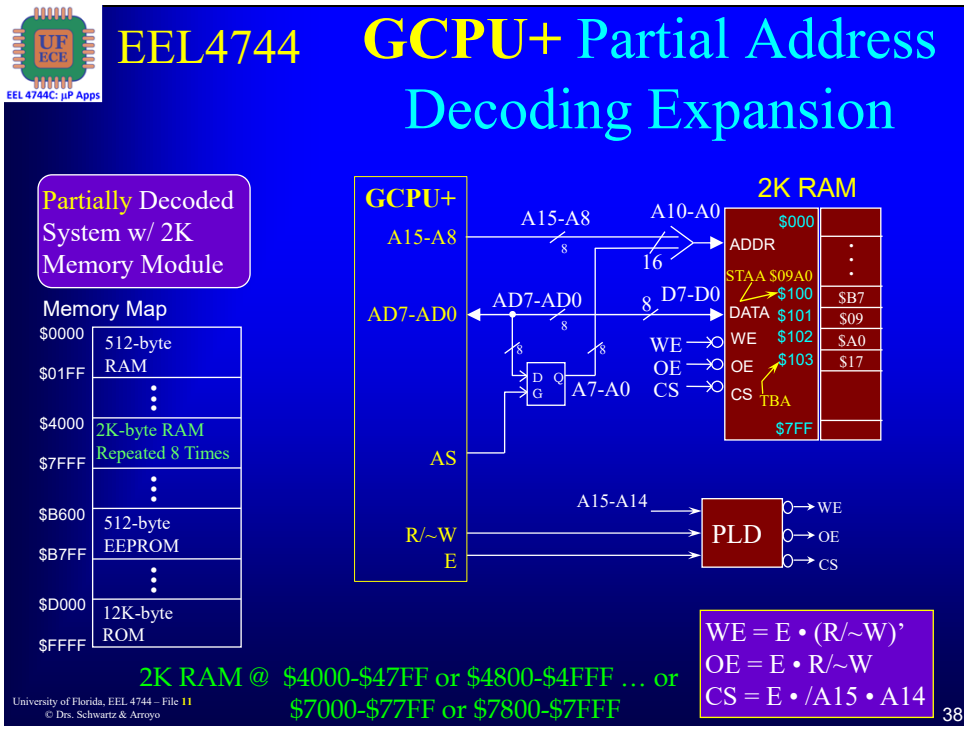
35


EEL4744
GCPU+ 32KB Memory Expansion
32K RAM
(at 0-$7FFF)
A14-A0
15
R/~W = R(H) = W(L)
W(L)
WE
GND
[or /R(H)]
OE
CS
8
D7-D0
E • /A15 • /Reset
• The output buffer is automatically disabled whenever WE is asserted, even if OE asserted
• I.e., WE has priority over OE.
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
36

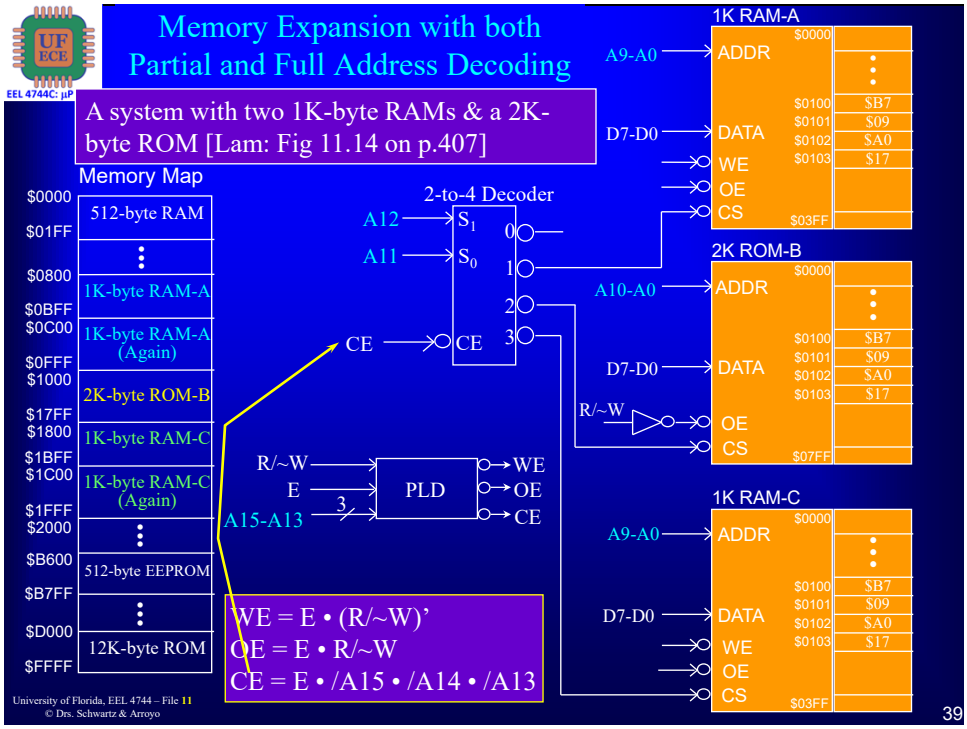
36


EEL4744
GCPU+ Full Address Decoding Expansion
Fully Decoded System w/ 2K Memory Module
Memory Map
$0000
512-byte RAM
$01FF
$4000
2K-byte RAM
$47FF
$B600
512-byte EEPROM
$B7FF
$D000
12K-byte ROM
$FFFF
GCPU+
A15-A8
AD7-AD0
AS
R/~W
E
A15-A8
A10-A0
16
AD7-AD0
8
D7-D0
A7-A0
WE
OE
CS
2K RAM
$000
ADDR
STAA $09A0
$100
DATA
$101
WE
$102
OE
$103
CS
TBA
$7FF
$B7
$09
$A0
$17
A15-A11
PLD
WE
OE
CS
WE = E • (R/~W)'
OE = E • R/~W
CS = E • /A15•A14•/A13•/A12•/A11
2K RAM @$4000-$47FF
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
37

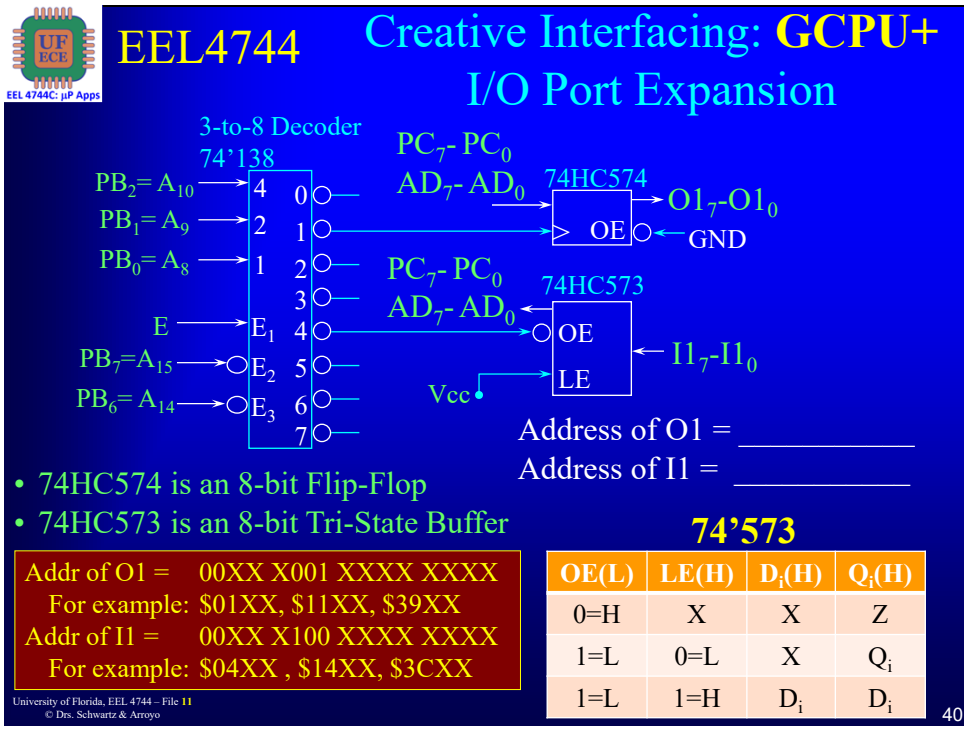
37


EEL4744
GCPU+ Partial Address Decoding Expansion
Partially Decoded System w/ 2K Memory Module
Memory Map
$0000
512-byte RAM
$01FF
$4000
2K-byte RAM Repeated 8 Times
$7FFF
$B600
512-byte EEPROM
$B7FF
$D000
12K-byte ROM
$FFFF
GCPU+
A15-A8
AD7-AD0
AS
R/~W
E
A15-A8
A10-A0
16
AD7-AD0
8
D7-D0
A7-A0
WE
OE
CS
2K RAM
$000
ADDR
STAA $09A0
$100
DATA
$101
WE
$102
OE
$103
CS
TBA
$7FF
$B7
$09
$A0
$17
A15-A14
PLD
WE
OE
CS
WE = E • (R/~W)'
OE = E • R/~W
CS = E • /A15 • A14
2K RAM @ $4000-$47FF or $4800-$4FFF … or
$7000-$77FF or $7800-$7FFF
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
38

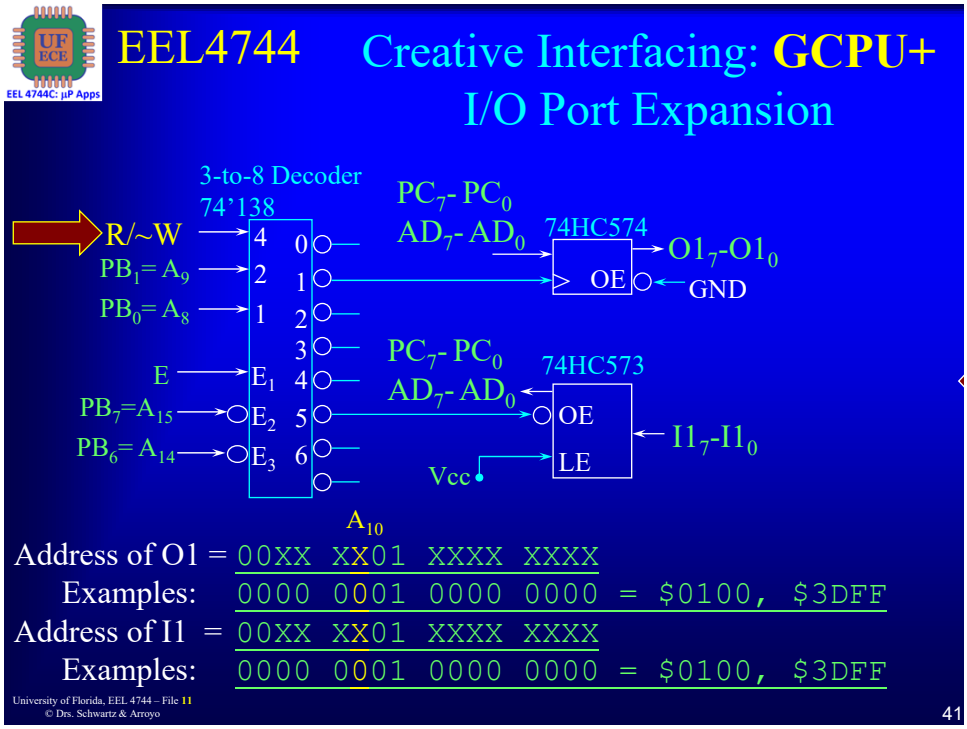
38


Memory Expansion with both Partial and Full Address Decoding
A system with two 1K-byte RAMs & a 2K-byte ROM [Lam: Fig 11.14 on p.407]
Memory Map
$0000 512-byte RAM $01FF
$0800 1K-byte RAM-A $0BFF
$0C00 1K-byte RAM-A (Again) $0FFF
$1000 2K-byte ROM-B $17FF
$1800 1K-byte RAM-C $1BFF
$1C00 1K-byte RAM-C (Again) $1FFF
$2000 $B600 512-byte EEPROM $B7FF
$D000 12K-byte ROM $FFFF
2-to-4 Decoder
A12 A11 CE
PLD
R/~W E A15-A13 3
WE OE CE
1K RAM-A
2K ROM-B
1K RAM-C
A9-A0 A10-A0 D7-D0 R/~W
ADDR DATA WE OE CS
$0000 $0100 $0101 $0102 $0103 $03FF $07FF
$B7 $09 $A0 $17
WE = E • (R/~W)'
OE = E • R/~W
CE = E • /A15 • /A14 • /A13
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
39

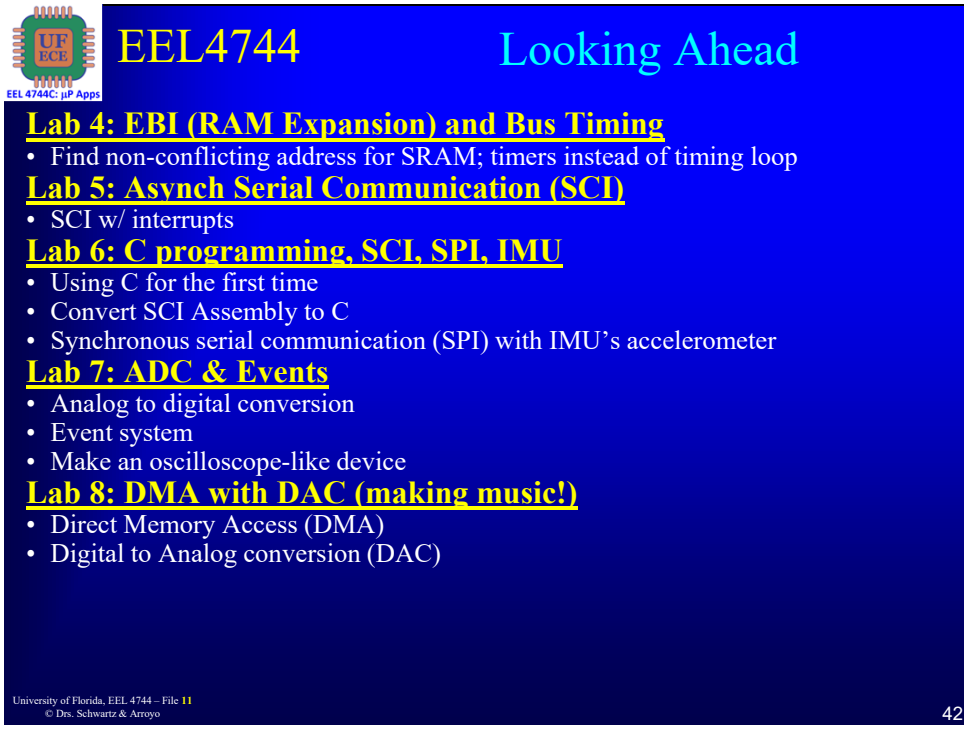
39


EEL4744
Creative Interfacing: GCPU+ I/O Port Expansion
3-to-8 Decoder 74'138
PB2= A10 PB1= A9 PB0= A8 E PB7=A15 PB6= A14
E1 E2 E3
0 1 2 3 4 5 6 7
PC7- PC0 AD7- AD0
74HC574
OE
O17-O10
GND
74HC573
LE
I17-I10
Vcc
Address of O1 = _____
Address of I1 = _____
• 74HC574 is an 8-bit Flip-Flop
• 74HC573 is an 8-bit Tri-State Buffer
Addr of O1 = 00XX X001 XXXX XXXX
For example: $01XX, $11XX, $39XX
Addr of I1 = 00XX X100 XXXX XXXX
For example: $04XX , $14XX, $3CXX
74'573
OE(L) LE(H) Di(H) Qi(H)
0=H X X Z
1=L 0=L X Qi
1=L 1=H Di Di
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
40

40


EEL4744
Creative Interfacing: GCPU+
I/O Port Expansion
3-to-8 Decoder
74'138
R/~W
PB1= A9
PB0= A8
E
PB7=A15
PB6= A14
A10
PC7- PC0
AD7- AD0
74HC574
OE
GND
O17-O10
PC7- PC0
AD7- AD0
74HC573
OE
LE
Vcc
I17-I10
Address of O1 = 00XX XX01 XXXX XXXX
Examples: 0000 0001 0000 0000 = $0100, $3DFF
Address of I1 = 00XX XX01 XXXX XXXX
Examples: 0000 0001 0000 0000 = $0100, $3DFF
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
41

41


EEL4744
Looking Ahead
Lab 4: EBI (RAM Expansion) and Bus Timing
• Find non-conflicting address for SRAM; timers instead of timing loop
Lab 5: Asynch Serial Communication (SCI)
• SCI w/ interrupts
Lab 6: C programming, SCI, SPI, IMU
• Using C for the first time
• Convert SCI Assembly to C
• Synchronous serial communication (SPI) with IMU's accelerometer
Lab 7: ADC & Events
• Analog to digital conversion
• Event system
• Make an oscilloscope-like device
Lab 8: DMA with DAC (making music!)
• Direct Memory Access (DMA)
• Digital to Analog conversion (DAC)
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
42

42

EEL4744

XMEGA's EBI: External Bus Interface

- Read **the entire** section 27 in doc8331!
- Many external device options are available
  - > Use the external memory address range
    - – 0x4000 through 0xFF FFFF
      - 0x0000-0x3FFF has 14 bits go through all values ➔ 2^{14} = 16k = 16,384
    - – 0x4000 through 0x4FFF is 2^{12} = 4k = 4096
      - 2^{12} because 12 bits go through all possible values
    - – 0x1 0000 through 0x1 FFFF is 2^{16} = 64k = 65,536
    - – Therefore, 0x4000 through 0xFF FFFF gives 64k*256 - 16k = $2^{16} \times 2^{8}$ – 16k = 2^{24} = 16M – 16k
- Note: Manual's cycle times are for **internal, not external SRAM**
  - > See Appendix 36 in doc8331 for external device timing

See doc8331, Section 27

See doc8385, Section 28

k = 2^{10}
M = 2^{20}

University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
43

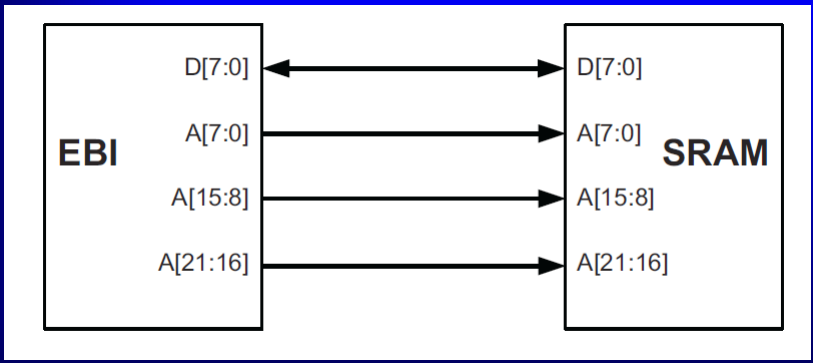
43

EEL4744

XMEGA's EBI: External Bus Interface

See doc8331, Figure 27-3

- Non-multiplexed SRAM connection



University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
44

44

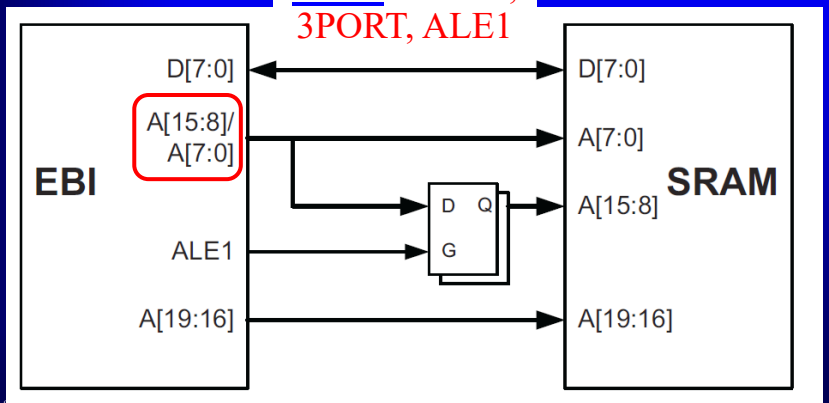
EEL4744

XMEGA's EBI: External Bus Interface

See doc8331, Figure 27-4

- Multiplexed SRAM connection using ALE1
  - > **Our µPad's with its Memory Base** uses **this** multiplexed expansion mode!

Mode: SRAM, 3PORT, ALE1



45

45

EEL4744

XMEGA's EBI: Pin-out for SRAM

See doc8331, Table 27-4

See doc8385, Sec 33.2, Tables 33-7, 33-8, 33-9

- Our uPad uses PortH (Port0), PortJ (Port1), and PortK (Port 2) as described below
  - > ALE's are **active-high**; some manuals are **wrong**

PORT	PIN	SRAM 3PORT ALE1		SRAM 3PORT ALE12	SRAM 4PORT ALE2	SRAM 4PORT NOALE
PORT3 (Port E or F)	7:0	–		–	A[15:8]	A[15:8]
PORT2 (Port K)	7:0	A[7:0]/ A[15:8]		A[7:0]/ A[15:8]/ A[23:16]	A[7:0]/ A[23:16]	A[7:0]
PORT1 (Port J)	7:0	D[7:0]		D[7:0]	D[7:0]	D[7:0]
	7:4	$\overline{CS}$[3:0] (A[19:16])	CS_1(L)	$\overline{CS}$[3:0]	$\overline{CS}$[3:0]	$\overline{CS}$[3:0] (A[21:18])
	3	–		ALE2	ALE2	A17
PORT0 (Port H)	2	ALE1 (H)		ALE1	–	A16
	1	RE (L)		RE	RE	RE
	0	WE (L)		WE	WE	WE

46

46

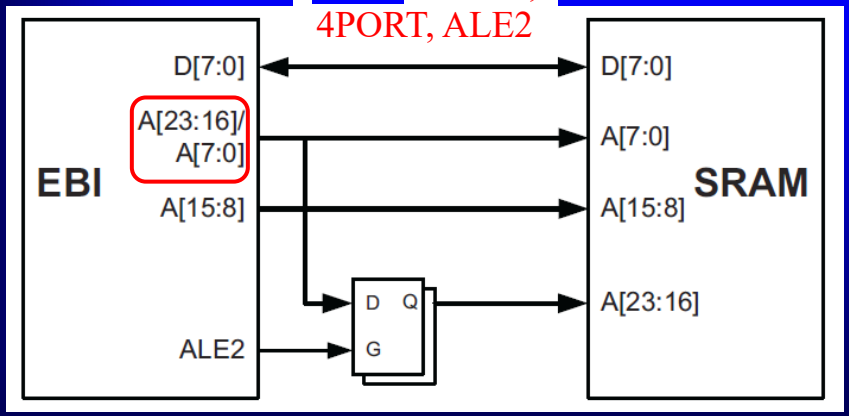
EEL4744

See doc8331, Figure 27-5

XMEGA's EBI: External Bus Interface

- Multiplexed SRAM connection using ALE2

Mode: SRAM, 4PORT, ALE2



47

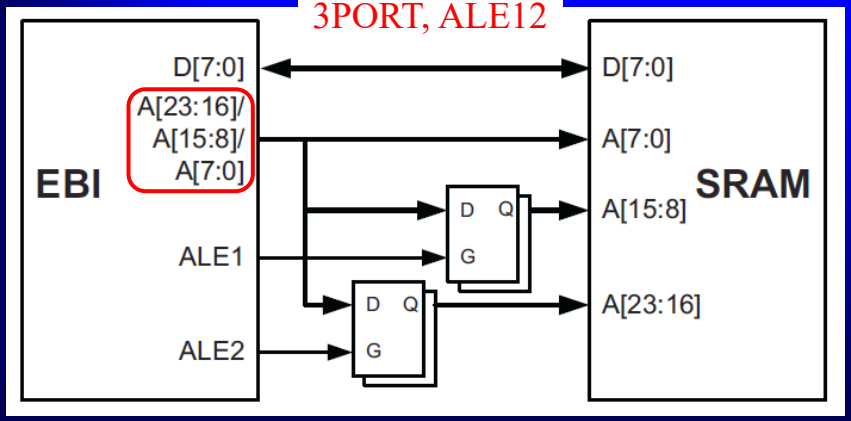
EEL4744

See doc8331, Figure 27-6

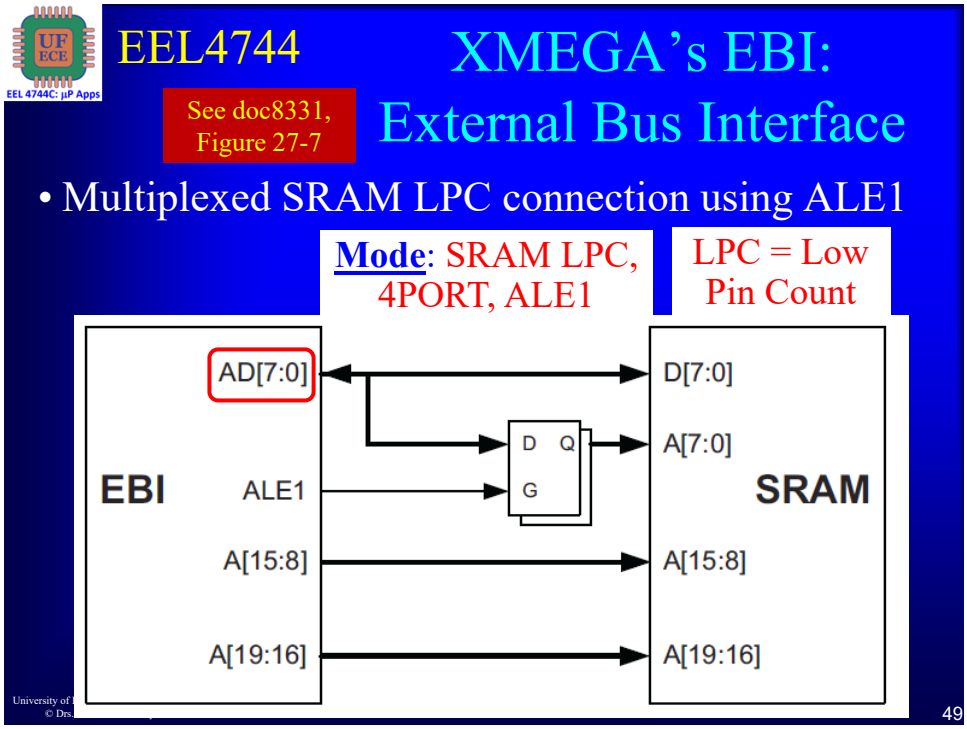
XMEGA's EBI: External Bus Interface

- Multiplexed SRAM connection using ALE1 and ALE2

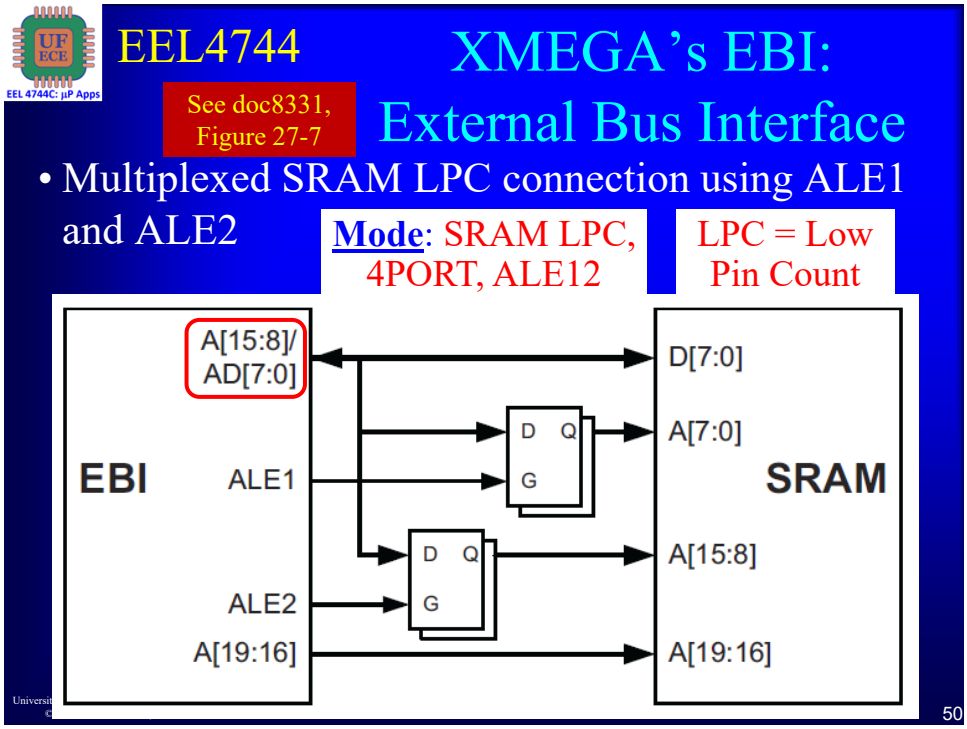
Mode: SRAM, 3PORT, ALE12



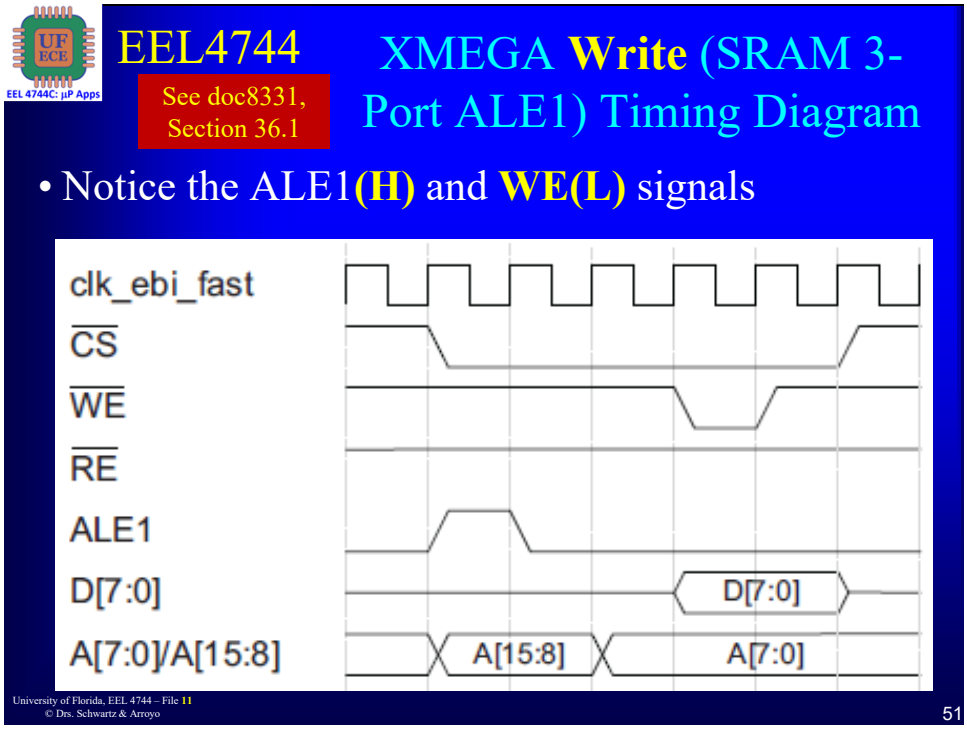
48


EEL4744
XMEGA's EBI:
External Bus Interface
See doc8331, Figure 27-7
• Multiplexed SRAM LPC connection using ALE1
Mode: SRAM LPC, 4PORT, ALE1
LPC = Low Pin Count
AD[7:0]
EBI
ALE1
A[15:8]
A[19:16]
D Q
G
D[7:0]
A[7:0]
SRAM
A[15:8]
A[19:16]
49

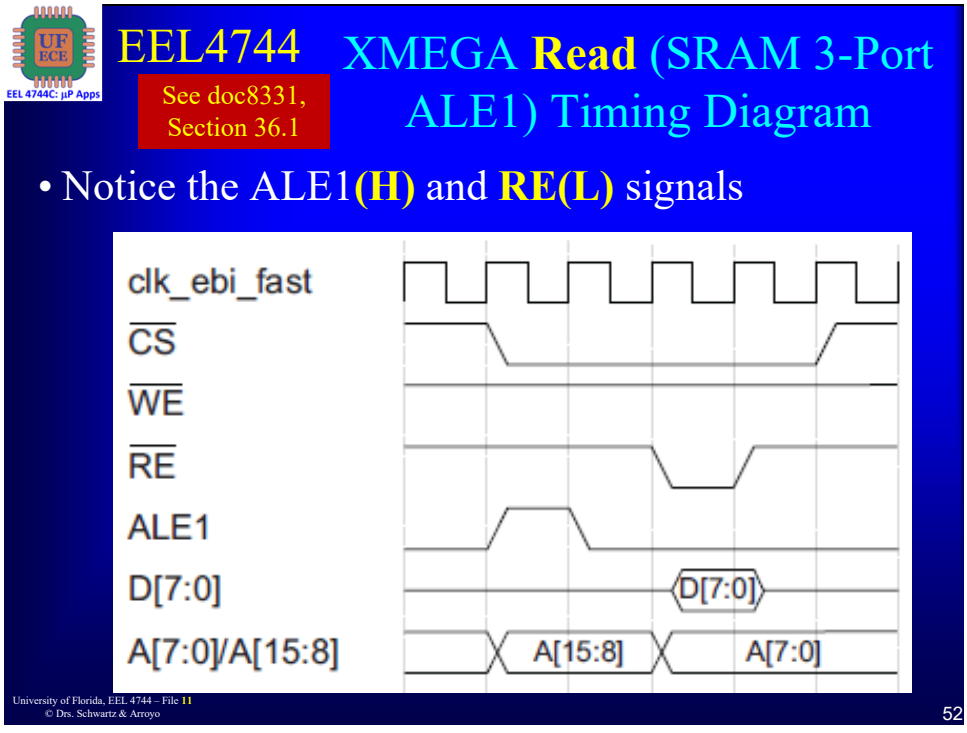
49


EEL4744
XMEGA's EBI:
External Bus Interface
See doc8331, Figure 27-7
• Multiplexed SRAM LPC connection using ALE1 and ALE2
Mode: SRAM LPC, 4PORT, ALE12
LPC = Low Pin Count
A[15:8]/ AD[7:0]
EBI
ALE1
ALE2
A[19:16]
D Q
G
D[7:0]
A[7:0]
SRAM
A[15:8]
A[19:16]
50

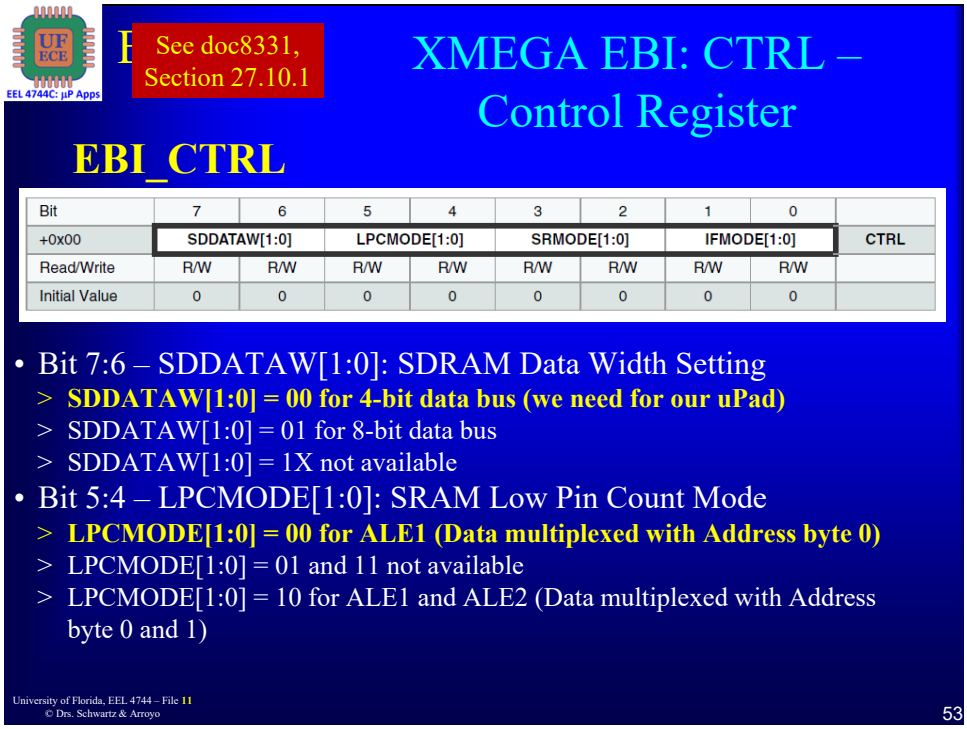
50


EEL4744
See doc8331, Section 36.1
XMEGA Write (SRAM 3-Port ALE1) Timing Diagram
• Notice the ALE1(H) and WE(L) signals
clk_ebi_fast
CS
WE
RE
ALE1
D[7:0]
A[7:0]/A[15:8]
D[7:0]
A[15:8]
A[7:0]
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
51

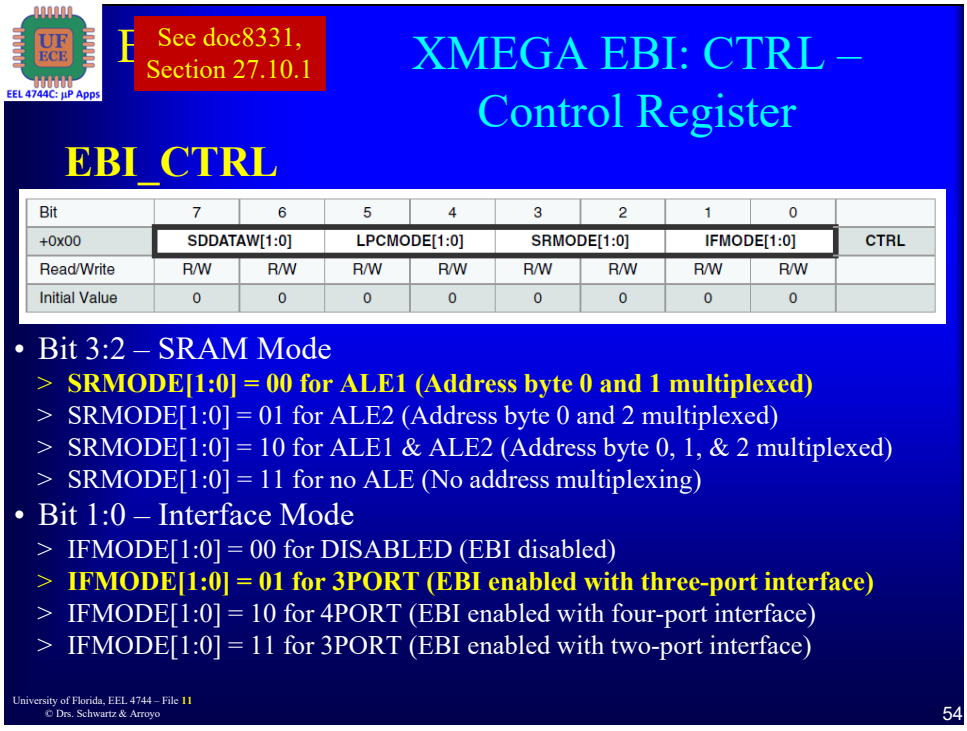
51


EEL4744
See doc8331, Section 36.1
XMEGA Read (SRAM 3-Port ALE1) Timing Diagram
• Notice the ALE1(H) and RE(L) signals
clk_ebi_fast
CS
WE
RE
ALE1
D[7:0]
A[7:0]/A[15:8]
D[7:0]
A[15:8]
A[7:0]
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
52

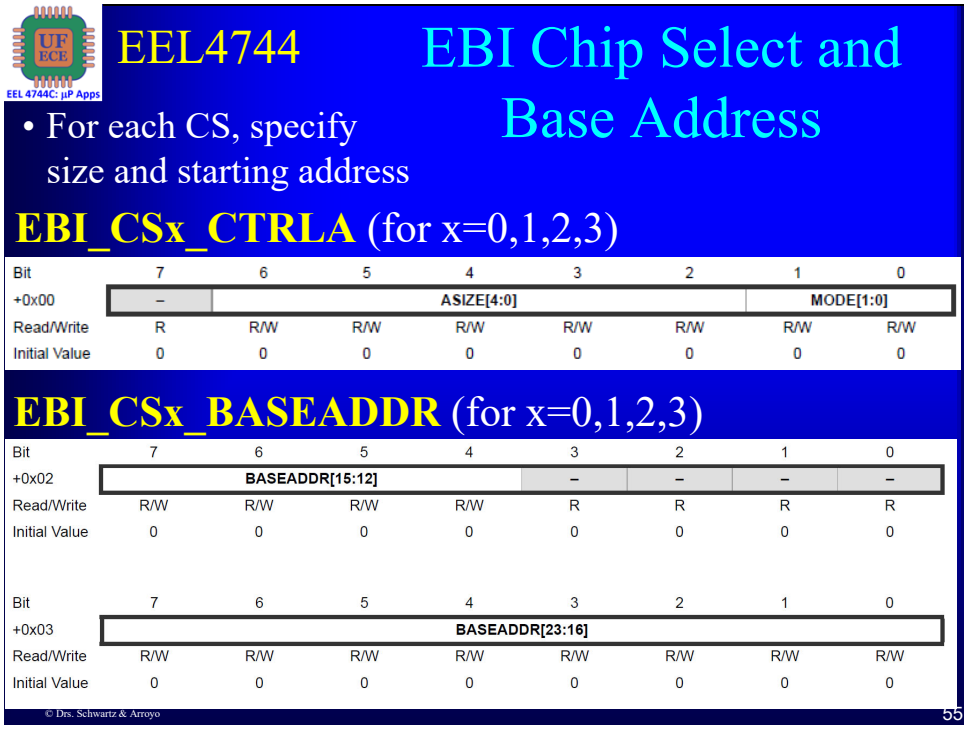
52


See doc8331, Section 27.10.1
XMEGA EBI: CTRL – Control Register
EBI_CTRL
Bit 7 6 5 4 3 2 1 0
+0x00 SDDATAW[1:0] LPCMODE[1:0] SRMODE[1:0] IFMODE[1:0] CTRL
Read/Write R/W R/W R/W R/W R/W R/W R/W R/W
Initial Value 0 0 0 0 0 0 0 0
• Bit 7:6 – SDDATAW[1:0]: SDRAM Data Width Setting
> SDDATAW[1:0] = 00 for 4-bit data bus (we need for our uPad)
> SDDATAW[1:0] = 01 for 8-bit data bus
> SDDATAW[1:0] = 1X not available
• Bit 5:4 – LPCMODE[1:0]: SRAM Low Pin Count Mode
> LPCMODE[1:0] = 00 for ALE1 (Data multiplexed with Address byte 0)
> LPCMODE[1:0] = 01 and 11 not available
> LPCMODE[1:0] = 10 for ALE1 and ALE2 (Data multiplexed with Address byte 0 and 1)
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
53

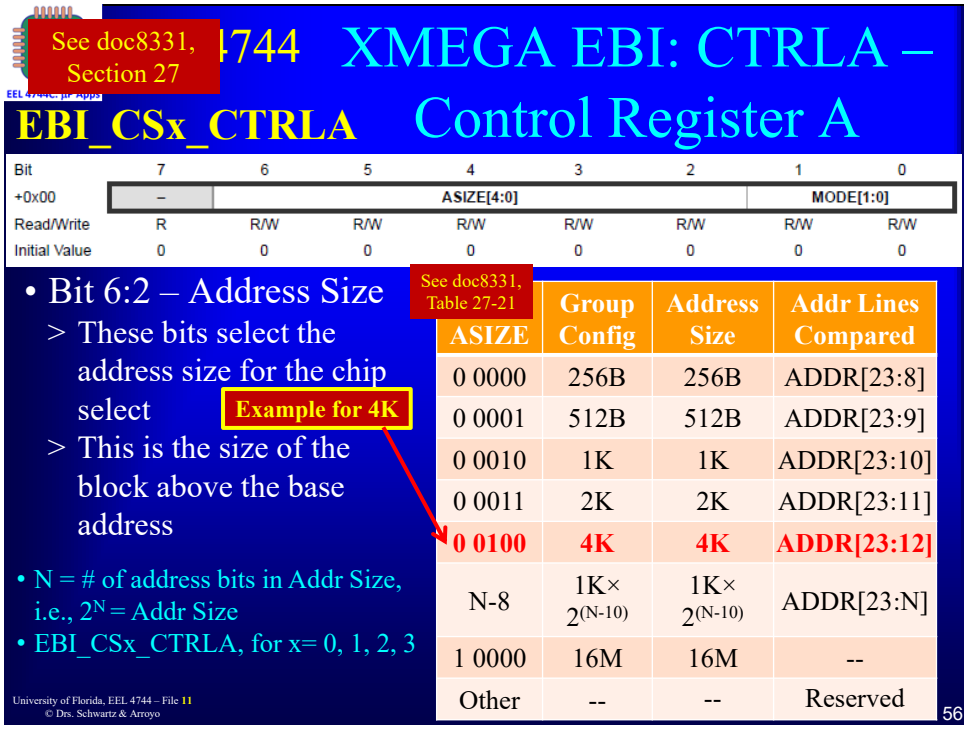
53


See doc8331, Section 27.10.1
XMEGA EBI: CTRL – Control Register
EBI_CTRL
Bit 7 6 5 4 3 2 1 0
+0x00 SDDATAW[1:0] LPCMODE[1:0] SRMODE[1:0] IFMODE[1:0] CTRL
Read/Write R/W R/W R/W R/W R/W R/W R/W R/W
Initial Value 0 0 0 0 0 0 0 0
• Bit 3:2 – SRAM Mode
> SRMODE[1:0] = 00 for ALE1 (Address byte 0 and 1 multiplexed)
> SRMODE[1:0] = 01 for ALE2 (Address byte 0 and 2 multiplexed)
> SRMODE[1:0] = 10 for ALE1 & ALE2 (Address byte 0, 1, & 2 multiplexed)
> SRMODE[1:0] = 11 for no ALE (No address multiplexing)
• Bit 1:0 – Interface Mode
> IFMODE[1:0] = 00 for DISABLED (EBI disabled)
> IFMODE[1:0] = 01 for 3PORT (EBI enabled with three-port interface)
> IFMODE[1:0] = 10 for 4PORT (EBI enabled with four-port interface)
> IFMODE[1:0] = 11 for 3PORT (EBI enabled with two-port interface)
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
54

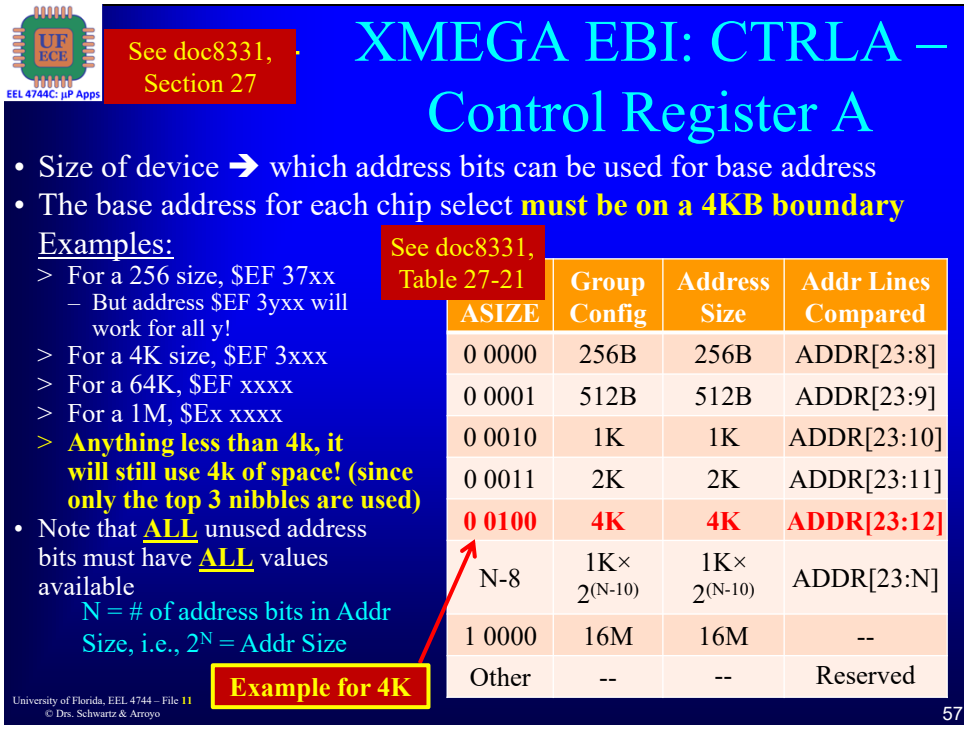
54


EEL4744
EBI Chip Select and Base Address
• For each CS, specify size and starting address
EBI_CSx_CTRLA (for x=0,1,2,3)
Bit 7 6 5 4 3 2 1 0
+0x00 – ASIZE[4:0] MODE[1:0]
Read/Write R R/W R/W R/W R/W R/W R/W R/W
Initial Value 0 0 0 0 0 0 0 0
EBI_CSx_BASEADDR (for x=0,1,2,3)
Bit 7 6 5 4 3 2 1 0
+0x02 BASEADDR[15:12] – – – –
Read/Write R/W R/W R/W R/W R R R R
Initial Value 0 0 0 0 0 0 0 0
Bit 7 6 5 4 3 2 1 0
+0x03 BASEADDR[23:16]
Read/Write R/W R/W R/W R/W R/W R/W R/W R/W
Initial Value 0 0 0 0 0 0 0 0
© Drs. Schwartz & Arroyo
55

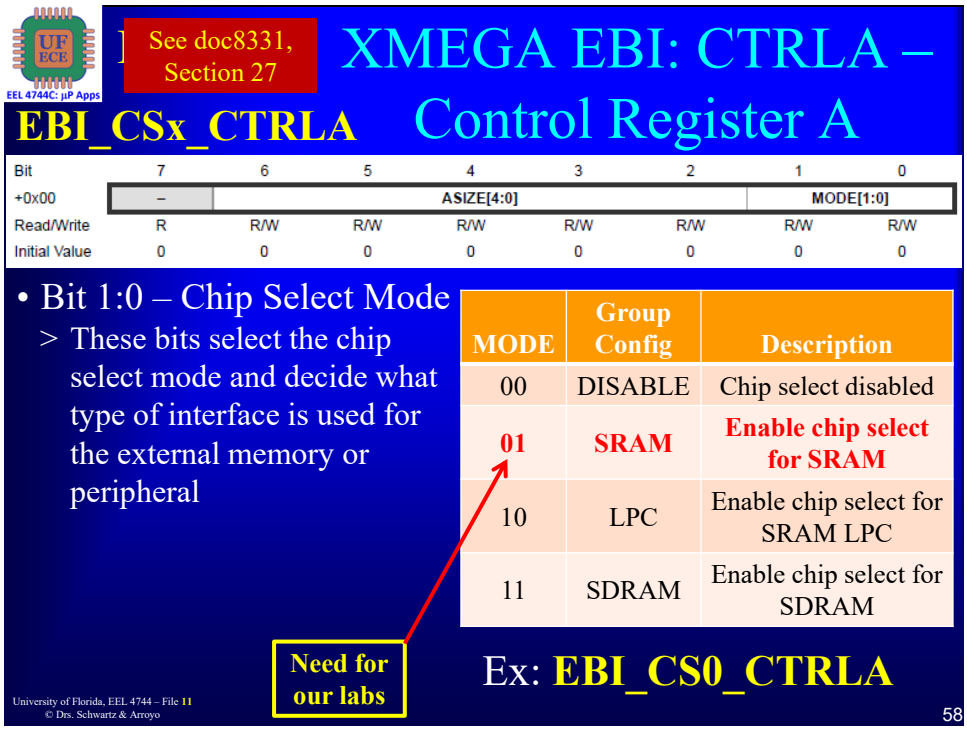
55


See doc8331, Section 27
4744
XMEGA EBI: CTRLA – Control Register A
EBI_CSx_CTRLA
Bit 7 6 5 4 3 2 1 0
+0x00 – ASIZE[4:0] MODE[1:0]
Read/Write R R/W R/W R/W R/W R/W R/W R/W
Initial Value 0 0 0 0 0 0 0 0
• Bit 6:2 – Address Size
> These bits select the address size for the chip select
> This is the size of the block above the base address
Example for 4K
• N = # of address bits in Addr Size, i.e., 2N = Addr Size
• EBI_CSx_CTRLA, for x= 0, 1, 2, 3
See doc8331, Table 27-21
ASIZE | Group Config | Address Size | Addr Lines Compared
0 0000 | 256B | 256B | ADDR[23:8]
0 0001 | 512B | 512B | ADDR[23:9]
0 0010 | 1K | 1K | ADDR[23:10]
0 0011 | 2K | 2K | ADDR[23:11]
0 0100 | 4K | 4K | ADDR[23:12]
N-8 | 1K× 2(N-10) | 1K× 2(N-10) | ADDR[23:N]
1 0000 | 16M | 16M | --
Other | -- | -- | Reserved
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
56

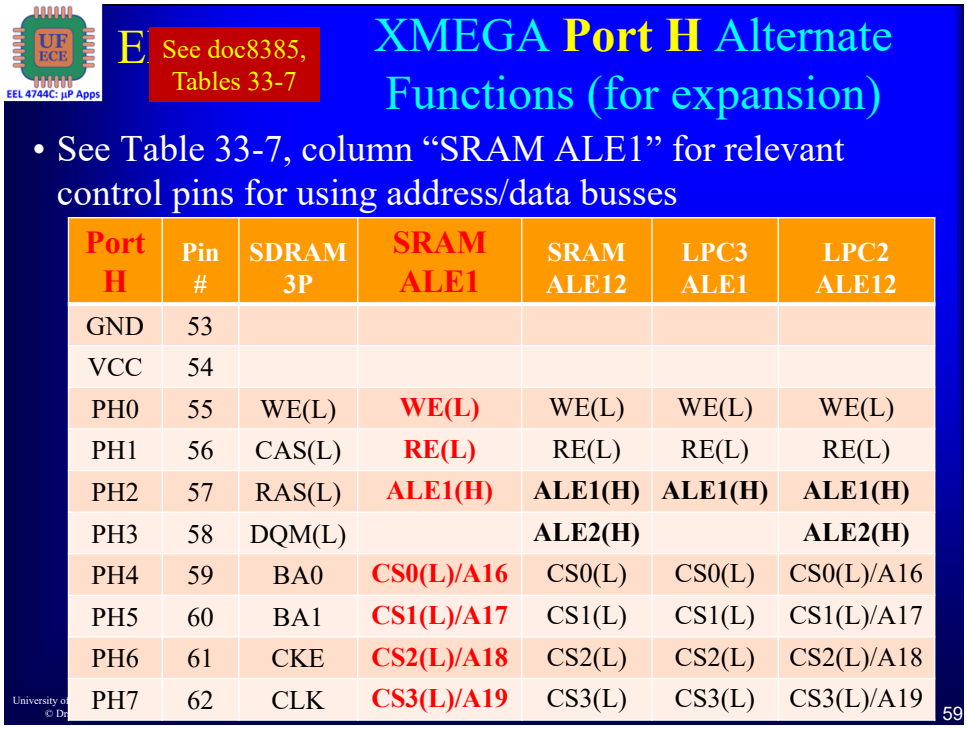
56


See doc8331, Section 27
XMEGA EBI: CTRLA – Control Register A
• Size of device ➔ which address bits can be used for base address
• The base address for each chip select must be on a 4KB boundary
Examples:
> For a 256 size, $EF 37xx
– But address $EF 3yxx will work for all y!
> For a 4K size, $EF 3xxx
> For a 64K, $EF xxxx
> For a 1M, $Ex xxxx
> Anything less than 4k, it will still use 4k of space! (since only the top 3 nibbles are used)
• Note that ALL unused address bits must have ALL values available
N = # of address bits in Addr Size, i.e., 2^N = Addr Size
See doc8331, Table 27-21
ASIZE | Group Config | Address Size | Addr Lines Compared
0 0000 | 256B | 256B | ADDR[23:8]
0 0001 | 512B | 512B | ADDR[23:9]
0 0010 | 1K | 1K | ADDR[23:10]
0 0011 | 2K | 2K | ADDR[23:11]
0 0100 | 4K | 4K | ADDR[23:12]
N-8 | 1K× 2^(N-10) | 1K× 2^(N-10) | ADDR[23:N]
1 0000 | 16M | 16M | --
Other | -- | -- | Reserved
Example for 4K
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
57

57

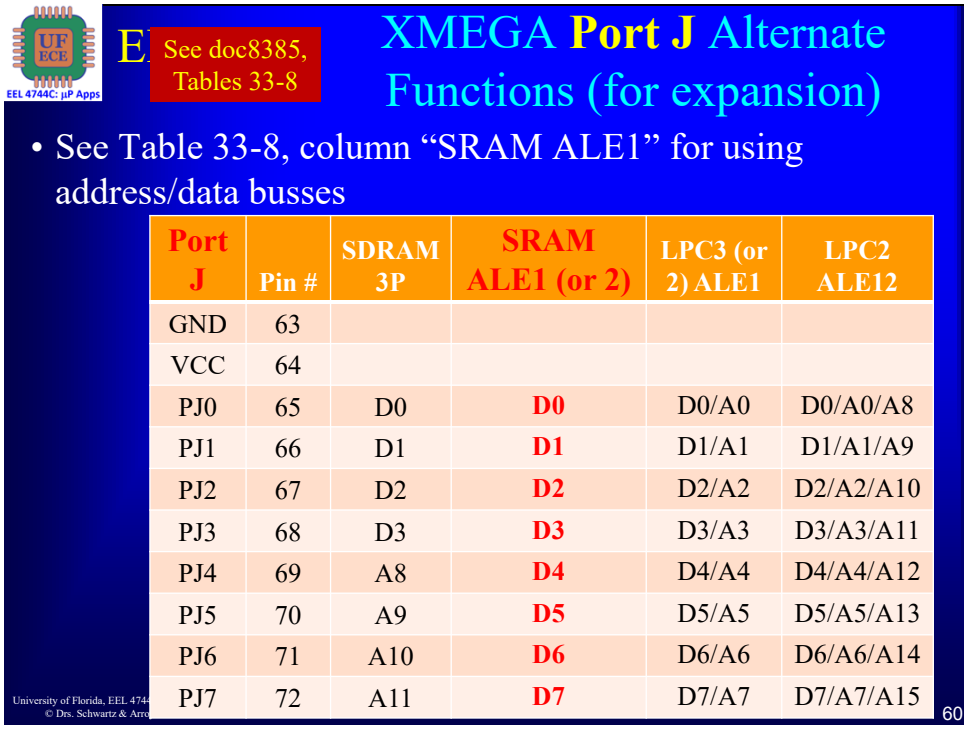

See doc8331, Section 27
XMEGA EBI: CTRLA – Control Register A
EBI_CSx_CTRLA
Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0
+0x00 | – | ASIZE[4:0] | | | | | MODE[1:0] |
Read/Write | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W
Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0
• Bit 1:0 – Chip Select Mode
> These bits select the chip select mode and decide what type of interface is used for the external memory or peripheral
MODE | Group Config | Description
00 | DISABLE | Chip select disabled
01 | SRAM | Enable chip select for SRAM
10 | LPC | Enable chip select for SRAM LPC
11 | SDRAM | Enable chip select for SDRAM
Need for our labs
Ex: EBI_CS0_CTRLA
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
58

58



Port H	Pin #	SDRAM 3P	SRAM ALE1	SRAM ALE12	LPC3 ALE1	LPC2 ALE12
GND	53					
VCC	54					
PH0	55	WE(L)	**WE(L)**	WE(L)	WE(L)	WE(L)
PH1	56	CAS(L)	**RE(L)**	RE(L)	RE(L)	RE(L)
PH2	57	RAS(L)	**ALE1(H)**	**ALE1(H)**	**ALE1(H)**	**ALE1(H)**
PH3	58	DQM(L)		**ALE2(H)**		**ALE2(H)**
PH4	59	BA0	**CS0(L)/A16**	CS0(L)	CS0(L)	CS0(L)/A16
PH5	60	BA1	**CS1(L)/A17**	CS1(L)	CS1(L)	CS1(L)/A17
PH6	61	CKE	**CS2(L)/A18**	CS2(L)	CS2(L)	CS2(L)/A18
PH7	62	CLK	**CS3(L)/A19**	CS3(L)	CS3(L)	CS3(L)/A19

59



Port J	Pin #	SDRAM 3P	SRAM ALE1 (or 2)	LPC3 (or 2) ALE1	LPC2 ALE12
GND	63				
VCC	64				
PJ0	65	D0	**D0**	D0/A0	D0/A0/A8
PJ1	66	D1	**D1**	D1/A1	D1/A1/A9
PJ2	67	D2	**D2**	D2/A2	D2/A2/A10
PJ3	68	D3	**D3**	D3/A3	D3/A3/A11
PJ4	69	A8	**D4**	D4/A4	D4/A4/A12
PJ5	70	A9	**D5**	D5/A5	D5/A5/A13
PJ6	71	A10	**D6**	D6/A6	D6/A6/A14
PJ7	72	A11	**D7**	D7/A7	D7/A7/A15

60

EEL 4744C: µP Apps

See doc8385, Tables 33-9

XMEGA Port K Alternate Functions (for expansion)

- See Table 33-9, column "SRAM ALE1" for using address bus

Port K	Pin #	SDRAM 3P	SRAM ALE1	SRAM ALE2	LPC3 ALE1
GND	73				
VCC	74				
PK0	75	A0	**A0/A8**	A0/A8/A16	A8
PK1	76	A1	**A1/A9**	A1/A9/A17	A9
PK2	77	A2	**A2/A10**	A2/A10/A18	A10
PK3	78	A3	**A3/A11**	A3/A11/A19	A11
PK4	79	A4	**A4/A12**	A4/A12/A20	A12
PK5	80	A5	**A5/A13**	A5/A13/A21	A13
PK6	81	A6	**A6/A14**	A6/A14/A22	A14
PK7	82	A7	**A7/A15**	A7/A15/A23	A15

61

EEL 4744C: µP Apps

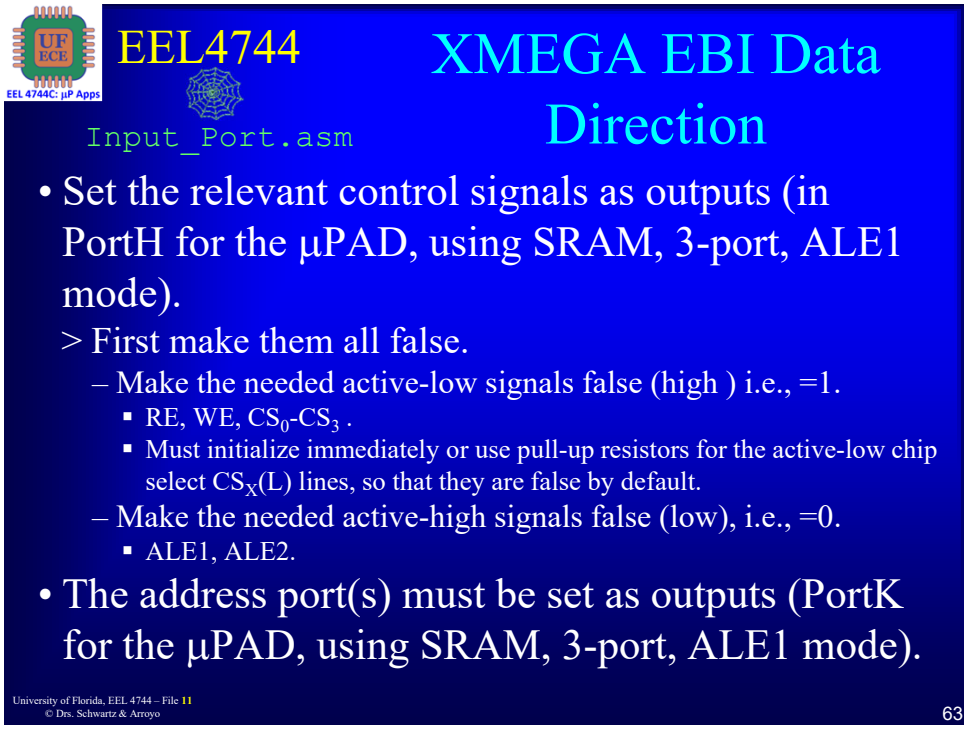
EEL4744

Input_Port.asm

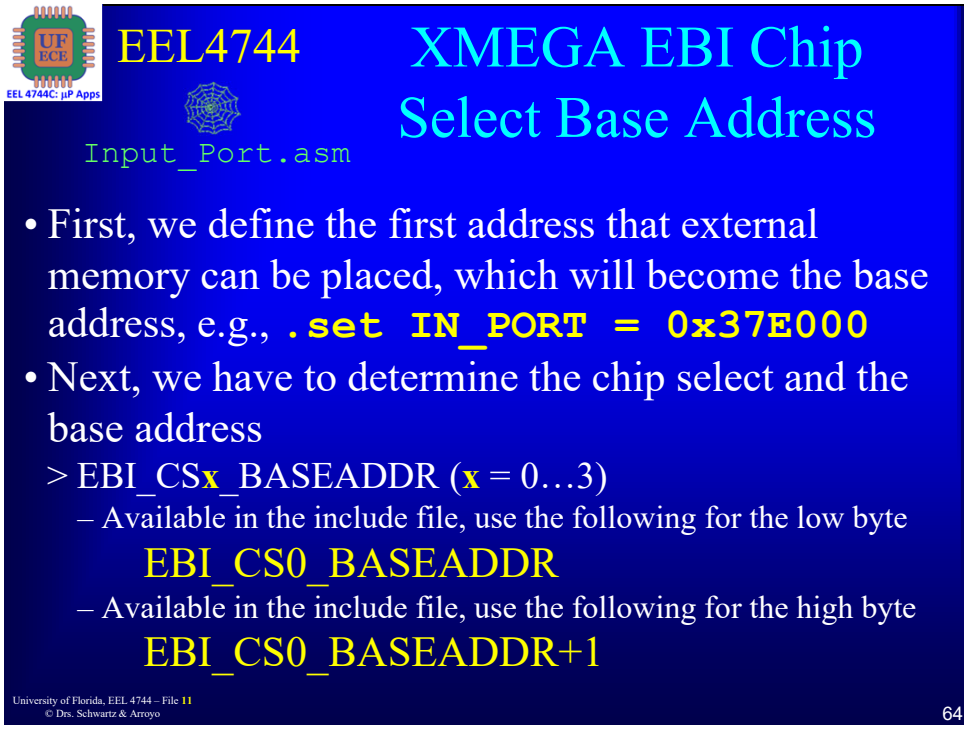
XMEGA EBI Chip Select Base Address

- For each CS, select a base address and a size to reserve an address range that will activate that CS.
- The base address will be used with the settings in EBI_CS**x**_CTRLA to determine a block of addresses for a specific chip select (**x** = 0, 1, 2, or 3).
- The base address has the following properties
  - > Consists of up to 12 (=24-n) bits for the address, A_{23}:A_n (n=12, 13, …).
    - – The lower n bits, A_{n-1}:A_0 , are assumed to be zero.

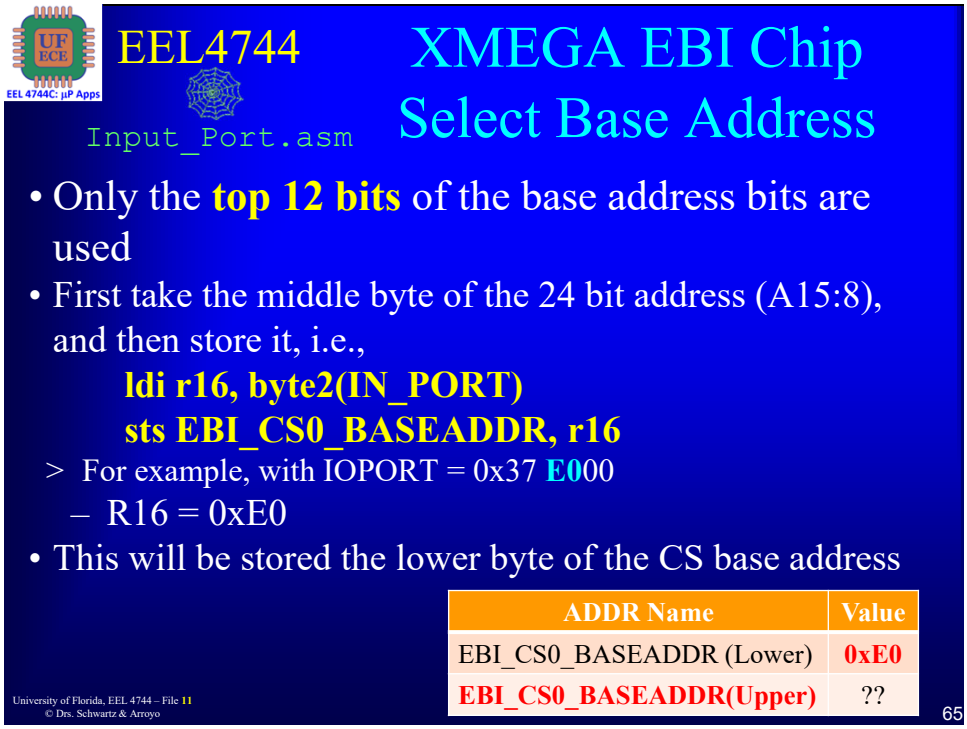
62


EEL4744
Input_Port.asm
XMEGA EBI Data Direction
• Set the relevant control signals as outputs (in PortH for the µPAD, using SRAM, 3-port, ALE1 mode).
> First make them all false.
– Make the needed active-low signals false (high) i.e., =1.
▪ RE, WE, CS0-CS3 .
▪ Must initialize immediately or use pull-up resistors for the active-low chip select CSX(L) lines, so that they are false by default.
– Make the needed active-high signals false (low), i.e., =0.
▪ ALE1, ALE2.
• The address port(s) must be set as outputs (PortK for the µPAD, using SRAM, 3-port, ALE1 mode).
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
63

63


EEL4744
Input_Port.asm
XMEGA EBI Chip Select Base Address
• First, we define the first address that external memory can be placed, which will become the base address, e.g., .set IN_PORT = 0x37E000
• Next, we have to determine the chip select and the base address
> EBI_CSx_BASEADDR (x = 0…3)
– Available in the include file, use the following for the low byte
EBI_CS0_BASEADDR
– Available in the include file, use the following for the high byte
EBI_CS0_BASEADDR+1
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
64

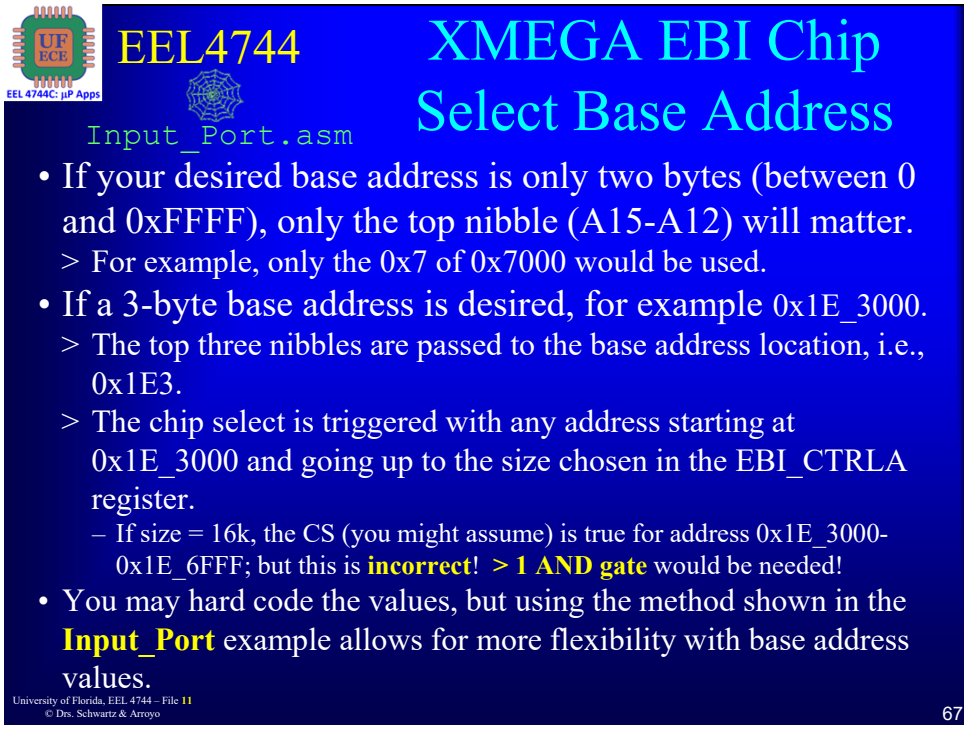
64


EEL4744
Input_Port.asm
XMEGA EBI Chip
Select Base Address
• Only the top 12 bits of the base address bits are used
• First take the middle byte of the 24 bit address (A15:8), and then store it, i.e.,
ldi r16, byte2(IN_PORT)
sts EBI_CS0_BASEADDR, r16
> For example, with IOPORT = 0x37 E000
– R16 = 0xE0
• This will be stored the lower byte of the CS base address
ADDR Name
Value
EBI_CS0_BASEADDR (Lower)
0xE0
EBI_CS0_BASEADDR(Upper)
??
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
65

65


EEL4744
Input_Port.asm
XMEGA EBI Chip
Select Base Address
• Next we shift the desired base address by 16 bits, load into a register, i.e.,
ldi r16, byte3(IN_PORT)
sts EBI_CS0_BASEADDR+1, r16
> For example, with IOPORT = 0x37 E000
– R16 = 0x37
ADDR Name
Value
EBI_CS0_BASEADDR (Lower)
0xE0
EBI_CS0_BASEADDR(Upper)
0x37
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
66

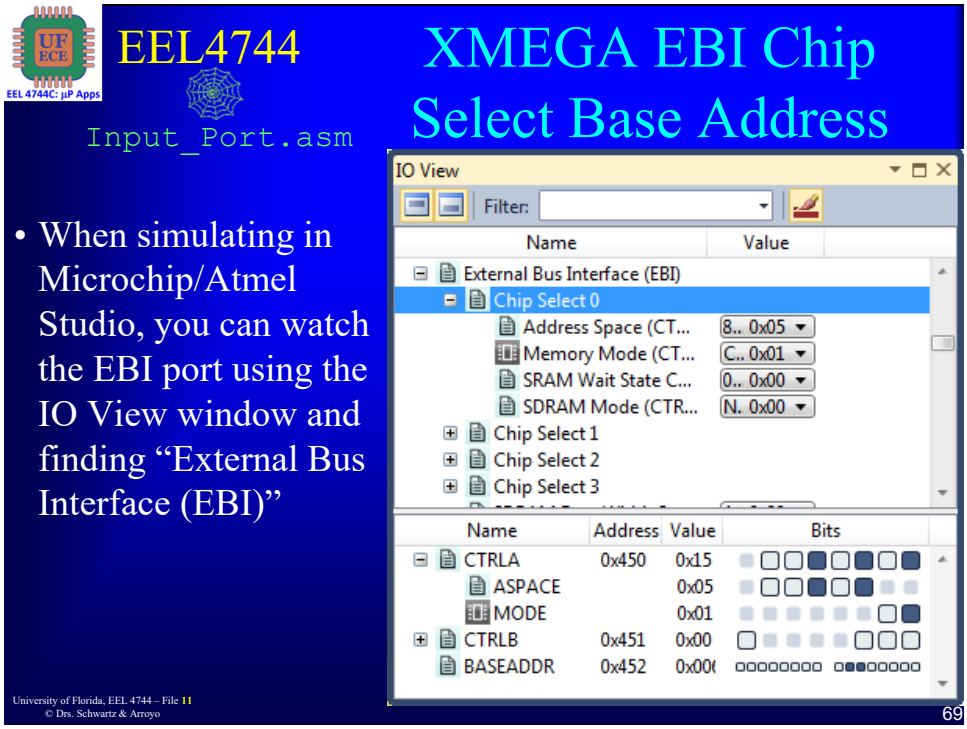
66


EEL4744
Input_Port.asm
XMEGA EBI Chip Select Base Address
• If your desired base address is only two bytes (between 0 and 0xFFFF), only the top nibble (A15-A12) will matter.
> For example, only the 0x7 of 0x7000 would be used.
• If a 3-byte base address is desired, for example 0x1E_3000.
> The top three nibbles are passed to the base address location, i.e., 0x1E3.
> The chip select is triggered with any address starting at 0x1E_3000 and going up to the size chosen in the EBI_CTRLA register.
– If size = 16k, the CS (you might assume) is true for address 0x1E_3000-0x1E_6FFF; but this is incorrect! > 1 AND gate would be needed!
• You may hard code the values, but using the method shown in the Input_Port example allows for more flexibility with base address values.
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
67

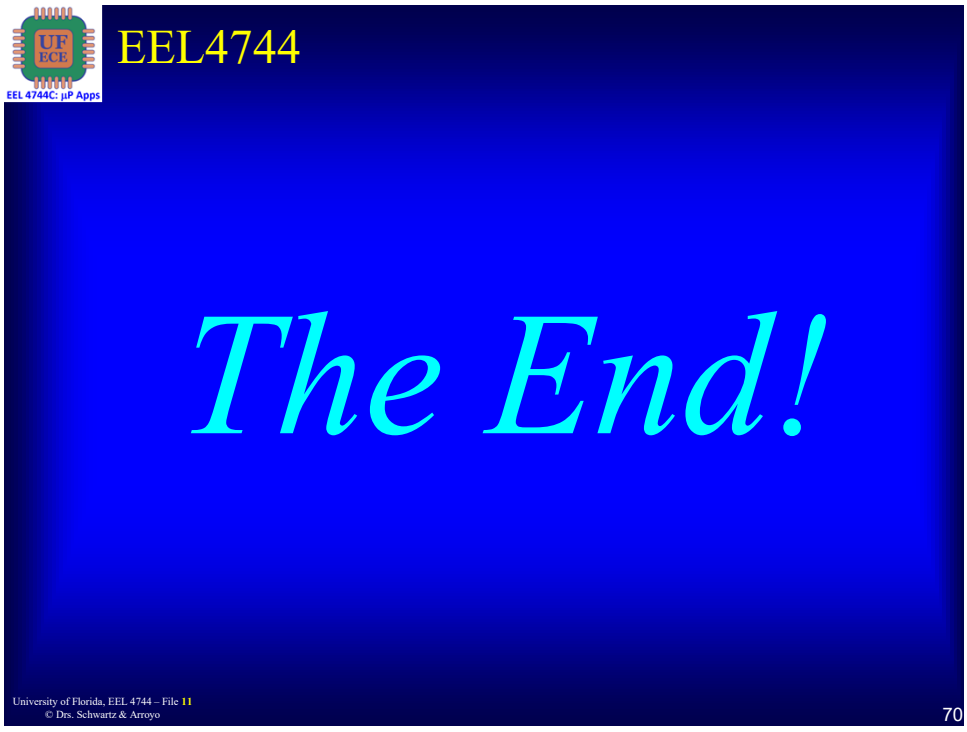
67


EEL4744
XMEGA EBI Chip Select (CS0-CS3)
• Each chip select must start at an "address boundary".
> The block size is determined by the address bits that can change.
– With the N changing address bits, the block size is 2^N.
> The other bits A23:AN must be fixed.
– A 4k block can start anywhere in expansion memory, e.g., 0x4000, 0x5000, 0x6000, … 0x1 0000, 0x1 1000, 0x1 2000, … 0x13 A000, …, 0x37 E000, …, 0xFF F000
▪ Only the least significant three hex digits (12 bit) can change (since 2^{12} = 4k).
– An 8k block can start at 0x4000, 0x6000, 0x8000, …
▪ It can NOT start at 0x5000, since the A15:A12 = 0101 or 0110
– A 16k block can start at 0x4000, 0x8000, 0xC000, …, 0x1 2000, …
▪ It can NOT start at 0x5000, 0x6000, 0x7000
▪ For a block starting at 0x4000, A15:A14=01
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
68

68


EEL4744
Input_Port.asm
XMEGA EBI Chip Select Base Address
• When simulating in Microchip/Atmel Studio, you can watch the EBI port using the IO View window and finding "External Bus Interface (EBI)"
IO View
Filter:
Name
Value
External Bus Interface (EBI)
Chip Select 0
Address Space (CT...
8.. 0x05
Memory Mode (CT...
C.. 0x01
SRAM Wait State C...
0.. 0x00
SDRAM Mode (CTR...
N. 0x00
Chip Select 1
Chip Select 2
Chip Select 3
Name
Address
Value
Bits
CTRLA
0x450
0x15
ASPACE
0x05
MODE
0x01
CTRLB
0x451
0x00
BASEADDR
0x452
0x00
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
69

69


EEL4744
The End!
University of Florida, EEL 4744 – File 11
© Drs. Schwartz & Arroyo
70

70