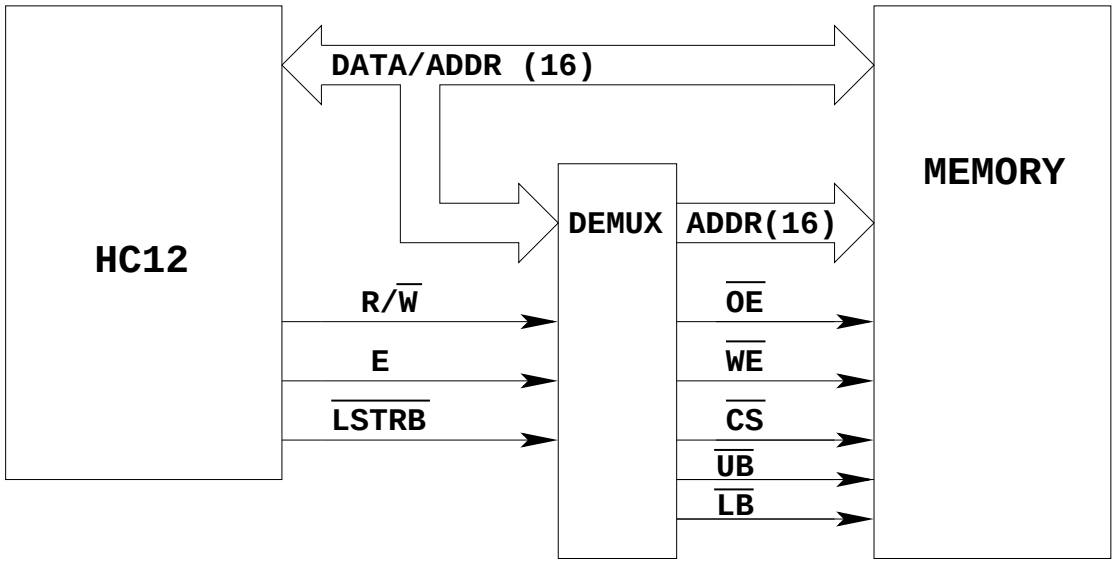
Multiplexed Address-Data Bus



$\overline{OE}$ – Output Enable (Read)
$\overline{WE}$ – Write Enable
$\overline{CS}$ – Chip Select
$\overline{UB}$ – Upper Byte Enable
$\overline{LB}$ – Lower Byte Enable

- The HC12 has a multiplexed address/data bus, AD15-0
- When in expanded mode the HC12 uses the pins for PORTA and PORTB as the multiplexed address/data bus
- When in expanded mode you cannot use PORTA or PORTB
- The HC12 can write one byte or two bytes in one memory cycle

Getting Into Expanded Mode

- The HC12 can operate in several modes:
  - Normal Single-Chip Mode (the way we have been using the HC12)
  - Normal Expanded Wide (16-bit address bus, 16-bit data bus)
  - Normal Expanded Narrow (16-bit address bus, 8-bit data bus)
  - Special Single Chip Mode (will not discuss)
  - Special Expanded Wide Mode (will not discuss)
  - Special Expanded Narrow Mode (will not discuss)
  - Special Peripheral Mode (will not discuss)
- How does the HC12 know what mode to run in?
- There are two ways:
  - When you reset the HC12 it looks at the state of three external pins: BKGD, MODB and MODA

    Based on the logic levels on these three pins the HC12 goes into one of the seven possible modes:

BKGD	MODB	MODA	Mode
0	0	0	Special Single Chip
0	0	1	Special Expanded Wide
0	1	0	Special Peripheral
0	1	1	Special Expanded Wide
1	0	0	Normal Single Chip
1	0	1	Normal Expanded Narrow
1	1	0	Reserved
1	1	1	Normal Expanded Wide

  - You can write to the MODE register to change the mode

    In Normal modes you can switch operating modes once by writing to the MODE register

    In Normal modes the MODE register is a write-once register

Coming out of reset the HC12 checks the state of the following three pins, and starts in one of several modes:

BKGD	MODB	MODA	
0	0	0	Special Single Chip
0	0	1	Special Expanded Wide
0	1	0	Special Peripheral
0	1	1	Special Expanded Wide
1	0	0	Normal Single Chip (Flash EEPROM on)
1	0	1	Normal Expanded Narrow
1	1	0	Reserved
1	1	1	Normal Expanded Wide (Flash EEPROM off)

On the EVBU, the HC12 starts in Normal Single Chip

After reset, can change modes by writing to MODE register

SMODN	MODB	MODA	ESTR	IVIS	EBSWAI	0	EME	MODE 0x000B
1	0	0	1	0	0	0	0	Reset (Normal Single Chip)
1	1	1	1	1	0	0	0	Need for Normal Expanded Wide

In normal modes can change MODE once.

To switch from Normal Single Chip to Normal Expanded Wide , write 11 to MODB:MODA.

Also turn on internal visibility by setting IVIS bit to 1

IVIS = 1 tells the HC12 to display internal bus cycles on the external bus

```
BSET $000B,#$68
```

Setting Up the HC12 For Expanded Mode

The PEAR Register

- We will start the HC12 in Normal Single Chip Mode, and write to the MODE register to switch it into Normal Expanded Wide Mode
- In Normal Expanded Mode the HC12 uses the E-clock and the R/$\overline{W}$ line for control of the external bus
- In Normal Single Chip Mode, the HC12 does not use these control lines, so it sets up their pins for general purpose I/O
- You need to tell the HC12 to drive these signals onto external pins in
- When enabled these signals are driven onto Port E order to use the HC12 in expanded mode
- You do this by writing to PEAR (Port E Assignment Register)
- PEAR is a write-once register

In Normal Single Chip mode, the E-Clock, LSTRB, and R_W lines are set up as general purpose I/O lines. When switched to Normal Expanded Wide need to make these control lines for expanded mode. Write to PEAR register:

NDBE	0	PIPOE	NECLK	LSTRE	RDWE	0	0	PEAR 0x000A
1	0	0	1	0	0	0	0	Reset Value (Single Chip)
0	0	1	0	1	1	0	0	Need for Expanded Wide

Write 0 to NECLK to enable E-Clock on external pin
Write 1 to LSTRE to enable LSTRB on external pin
Write 1 to RDWE to enable R_W on external pin
Write 1 to PIPOE to indicate state of instruction queue on external pins
Write 0 to NDBE to drive Data Bus Enable onto extenal pin

```
LDAA #$2C
STAA $000A
```

The MISC Register

- The MISC (Miscellaneous) register needs to be set up to allow the HC12 to operate properly in Expanded Mode
- The MISC register allows you to change the number of clock streches used by the HC12
- The MISC register allows you to disable the Flash EEPROM
- The MISC register allows you to move the Flash EEPROM to address block 0x0000 to 0x7fff
- When reset the HC12 forces three E-clock stretches to external data accesses
- This is useful when using slow memory devices
- We will use a fast memory device, so we will speed up external data accesses by forcing the HC12 to use no E-clock stretches
- We will leave the Flash EEPROM enabled at its normal address block of 0x8000 to 0xffff

0	NDRF	RFSTR1	RFSTR0	**EXSTR1**	**EXSTR0**	**MAPROM**	**ROMON**	**MISC 0x0013**
0	0	0	**0**	**1**	**1**	**1**	**1**	**Reset Value (Single Chip)**
0	0	0	**0**	**0**	**0**	**1**	**1**	**Need for Expanded Wide**

```
Change EXSTR1:EXSTR0 from 11 to 00 to change from three
E-clock stretches to no E-clock strech.

Leave MAPROM at 1 and ROMON at 1 to have DBug12 at 0x8000-0xFFFF

            BLCR $0013,#$0C
```

Putting the HC12 into Normal Expanded Wide Mode

- To put the HC12 into Normal Expanded Wide you must write to the MODE, PEAR and MISC registers
- When D-Bug12 runs, it writes to these registers as part of its start-up code
- Because these three registers are write-once, you must write to them before DBug-12 does
- In order to switch to Normal Expanded Wide mode, we will put code to write to these registers in EEPROM and set the jumpers on the EVBU to run out of EEPROM
- After writing to these registers we will jump into the normal DBug-12 code
- DBug-12 will try to write to these registers, but this will have no effect on these write-once registers


```
PEAR     equ   $000A       ; PEAR address
MODE     equ   $000B       ; MODE address
MISC     equ   $0013       ; MISC address
DBUG12   equ   $F700       ; Start of DBug-12

         org   $0D00

         MOVB  #$2C,PEAR   ; Enable LSTRB and R/W
         BSET  MODE,#$68   ; Expanded wide mode, turn on internal visibility
         BCLR  MISC,#$0C   ; Put in zero E-Clock stretches
         JMP   DBUG12      ; Jump to DBug-12
```

One-Byte and Two-Byte Data Accesses in Normal Expanded Wide Mode

- In Normal Expanded Wide Mode the HC12 can access one byte or two bytes in a single memory cycle
- A single byte can be located at an even address or an odd address
- The data lines for bytes at even addresses are connected to AD15-8 lines
  - Because the data for even bytes are accessed through the upper 8 bits of the data bus, these are called Upper Bytes (or High Bytes)
- The data lines for bytes at odd addresses are connected to AD7-0 lines
  - Because the data for even bytes are accessed through the lower 8 bits of the data bus, these are called Low Bytes
- The HC12 can access a High Byte and the following Low Byte in one memory access
  - For such a 16-bit access the 15 upper address lines are the same; only the lower address line is different
- The HC12 cannot access a Low Byte and the following High Byte
  - The 15 upper address bits are different for these bytes, and it would be difficult to build a decoder which could deal with this
  - For example, the addresses of the bytes at 0x7fff and 0x8000 are:

```
011111111111111   0x7fff
100000000000000   0x8000
```

One-Byte and Two-Byte Data Accesses in Normal Expanded Wide Mode

- To determine whether the HC12 is trying to access a single High Byte, a single Low Byte, or two bytes (High Byte and following Low Byte) the HC12 uses the A0 address line and the $\overline{\text{LSTRB}}$ control line
- $\overline{\text{LSTRB}}$ = 0 means access low (odd) byte
- A0 = 0 means access high (even) byte

$\overline{\text{LSTRB}}$	A0	Access Type
1	0	8-bit access of even address (high byte)
0	1	8-bit access of odd address (low byte)
0	0	16-bit access of even address (high and low byte)
1	1	Cannot occur

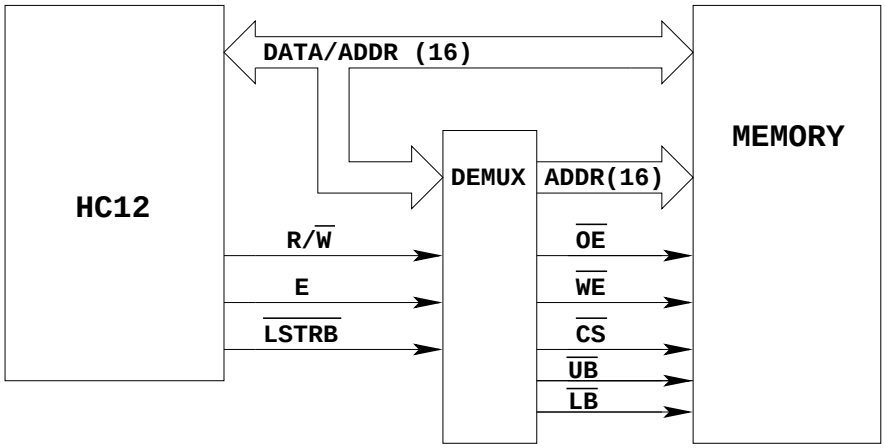
- Note: $\overline{\text{LSTRB}}$ = 1 and A0 = 1 would imply access of low (odd) byte and following high (even) byte. This will not occur for external data accesses on the HC12
- If the HC12 needs to access a 16-bit number from an odd address, it will do this in two memory cycles — it will access the 8-bit number at the odd address, followed by the 8-bit number at the even address
- For example the instruction

```
movw #$55aa,$08ff
```

  will write the 0x55 to memory location 0x08ff on one memory cycle, and the 0xaa to memory location 0x0900 on the next memory cycle.

The HC12 in Expanded Wide Mode

- In expanded mode the HC12 can communicate with external memory and devices over the multiplexed address/data bus
- To connect an external device to the HC12 bus you need to identify an unused region of memory
- We will map an external memory chip to address block 0x1000 to 0x7FFF
- We can map other devices into the unused memory from 0x0400 to 0x07FF and 0x0C00 to 0x0CFF
- Need decoder to demultiplex address from data, and to generate control lines needed by memory



$\overline{OE}$ – Output Enable (Read)
$\overline{WE}$ – Write Enable
$\overline{CS}$ – Chip Select
$\overline{UB}$ – Upper Byte Enable
$\overline{LB}$ – Lower Byte Enable

Want to map memory to address block 0x1000 to 0x7fff

How can we tell when an address is within this range?

The 12 LSB of the address can be anything (don't care)

The 4 MSB of the address must be 0001, 0010, 0011,
0100, 0101, 0110, 0111

A15 must be 0, one of A14, A13, A12 must be non-zero

$\overline{CS}$ = !(!A15 & (A14 # A13 # A12) & E)

Want memory to drive data bus when R/W = 1 and E = 1

$\overline{OE}$ = !(RW & E)

Want to write to memory when R/W = 0 and E = 1

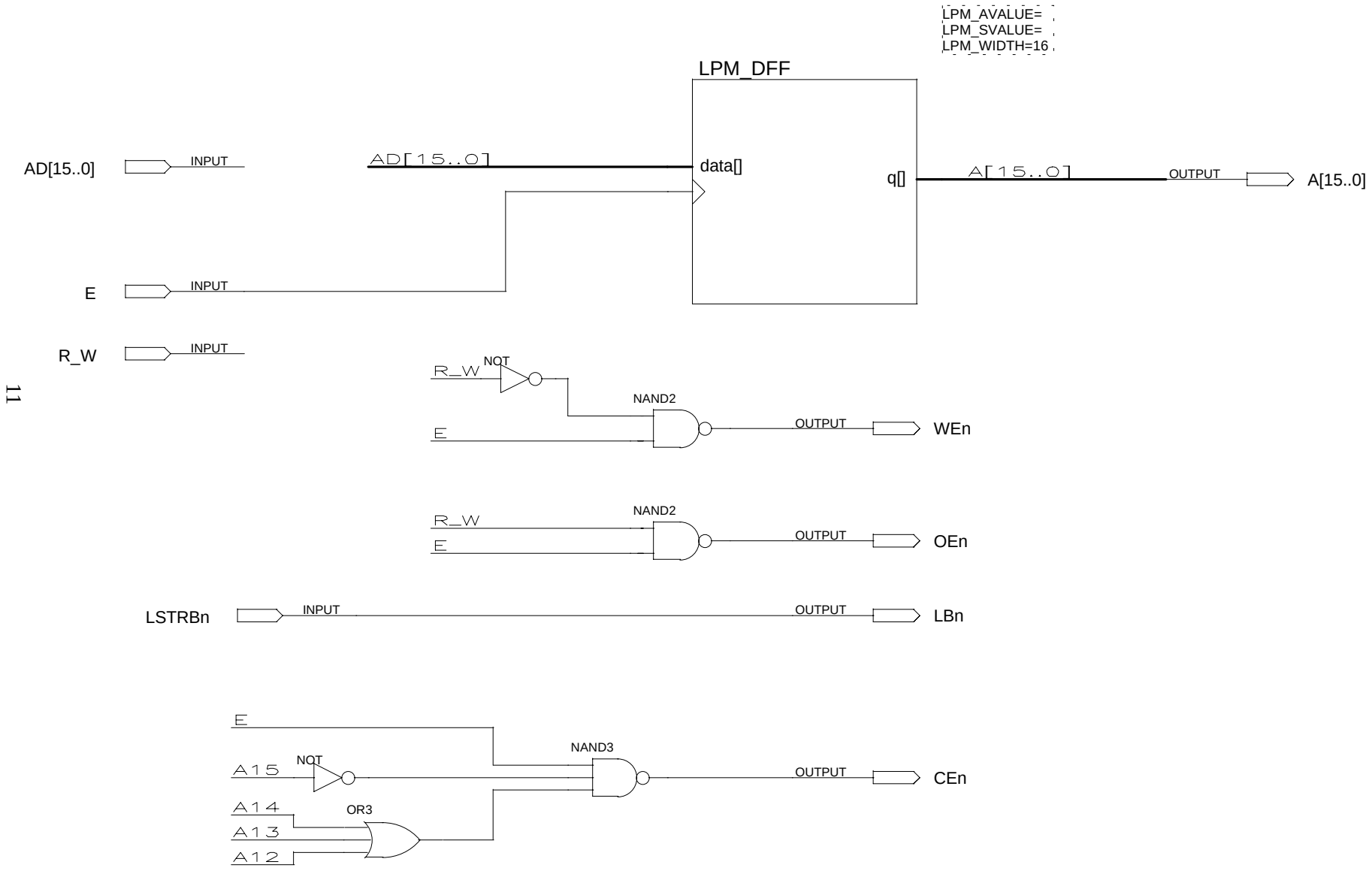
$\overline{WE}$ = !(!RW & E)

Want to access lower (odd) byte when LSTRB = 0

$\overline{LB}$ = $\overline{LSTRB}$

Want to access upper (even) byte when A0 = 0

$\overline{UB}$ = A0


LPM_AVALUE=
LPM_SVALUE=
LPM_WIDTH=16
LPM_DFF
AD[15..0]
INPUT
AD[15..0]
data[]
q[]
A[15..0]
OUTPUT
A[15..0]
E
INPUT
R_W
INPUT
R_W
NOT
NAND2
E
OUTPUT
WEn
R_W
NAND2
E
OUTPUT
OEn
LSTRBn
INPUT
OUTPUT
LBn
E
A15
NOT
NAND3
A14
A13
A12
OR3
OUTPUT
CEn

Timing on the Expanded Bus