SISTEMI EMBEDDED
AA 2013/2014

SOPC DE2 Basic Computer
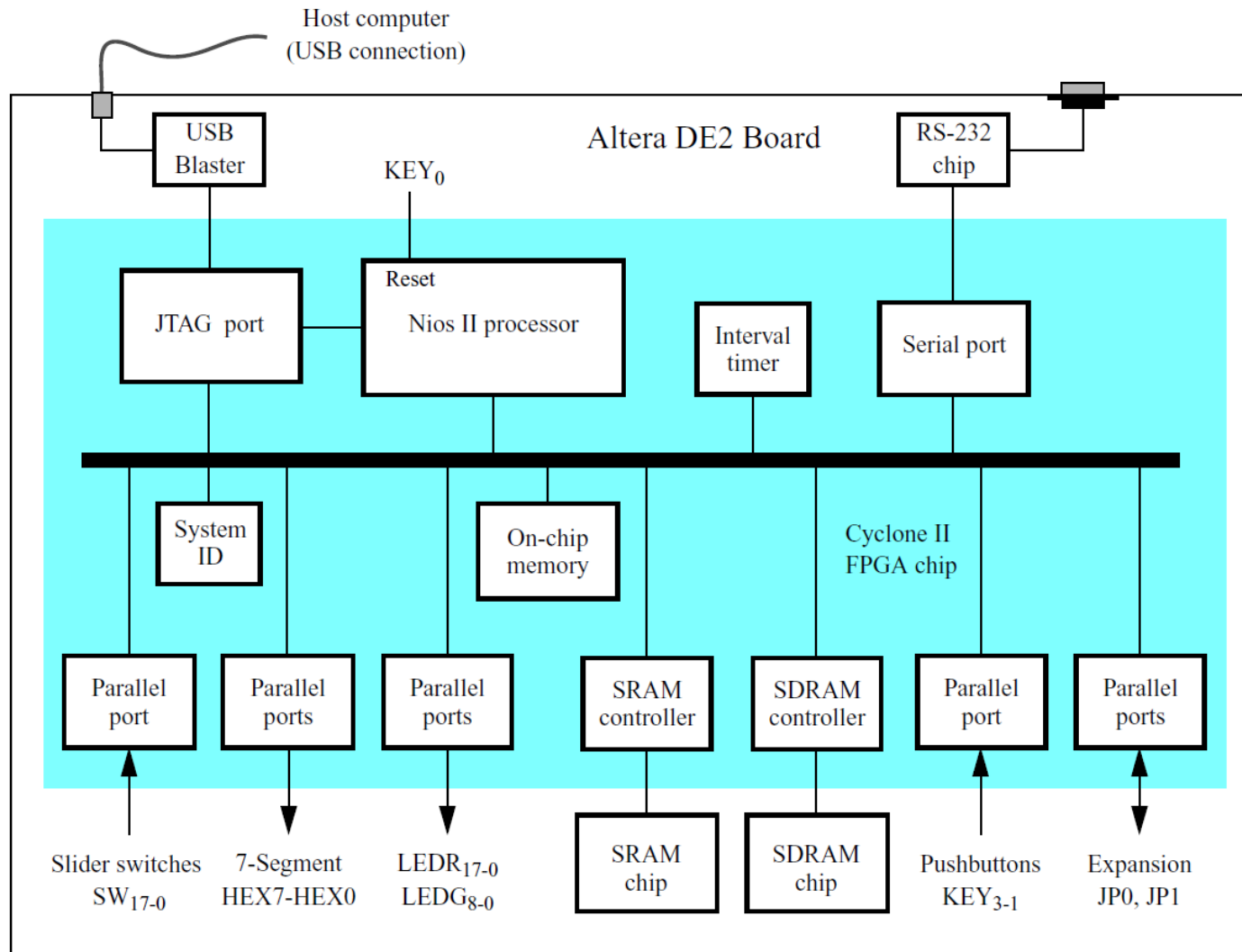
Parallel port

Federico Baronti

DE2 Basic Computer

- Nios II configuration provided by Altera University Program
  - Core: Nios II/e
  - Memory: SDRAM, RAM, On-chip memory
  - Parallel ports: Red_LEDs, Green_LEDs, HEX3_HEX0, HEX7_HEX4, Slider_switches, Pushbuttons, etc.
  - Other peripherals: JTAG UART, Serial_port, Interval_timer, sysid

DE2 Basic Computer (cont.)

Parallel port (1)

- Interface for general purpose I/O
  - Based on Altera's PIO core customized for DE-series boards
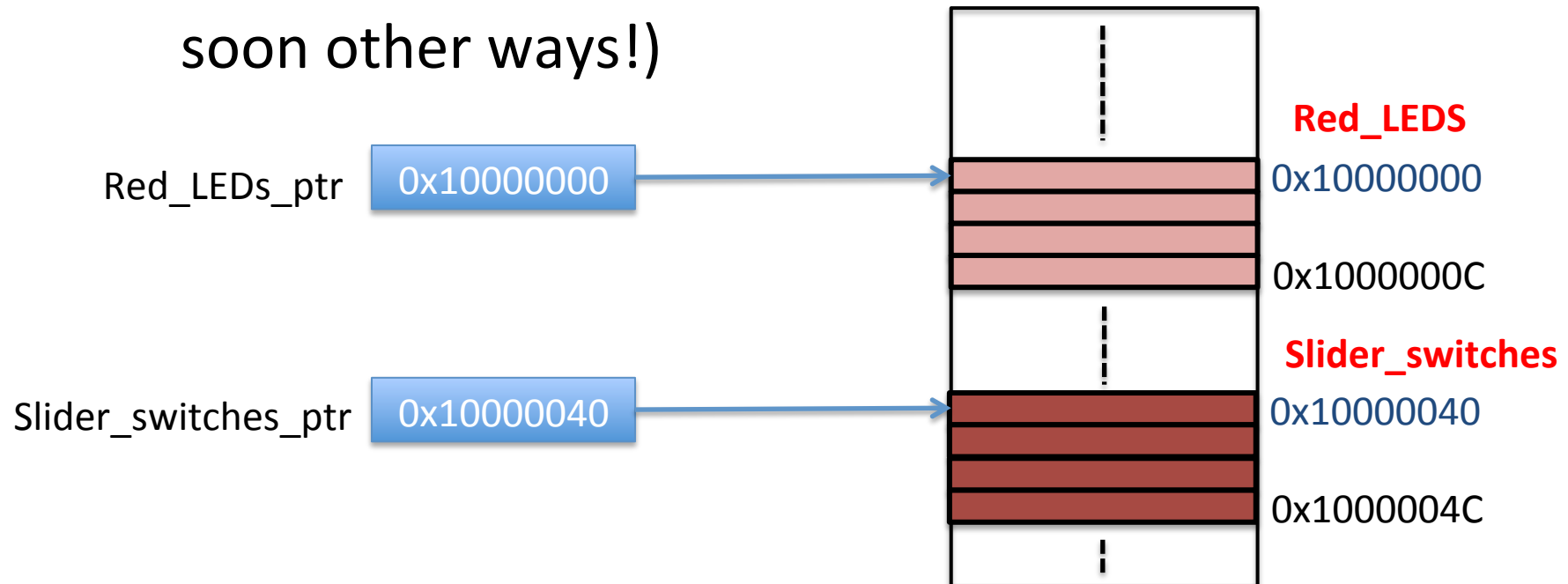 - Controlling LEDs, acquiring data from Switches, etc.
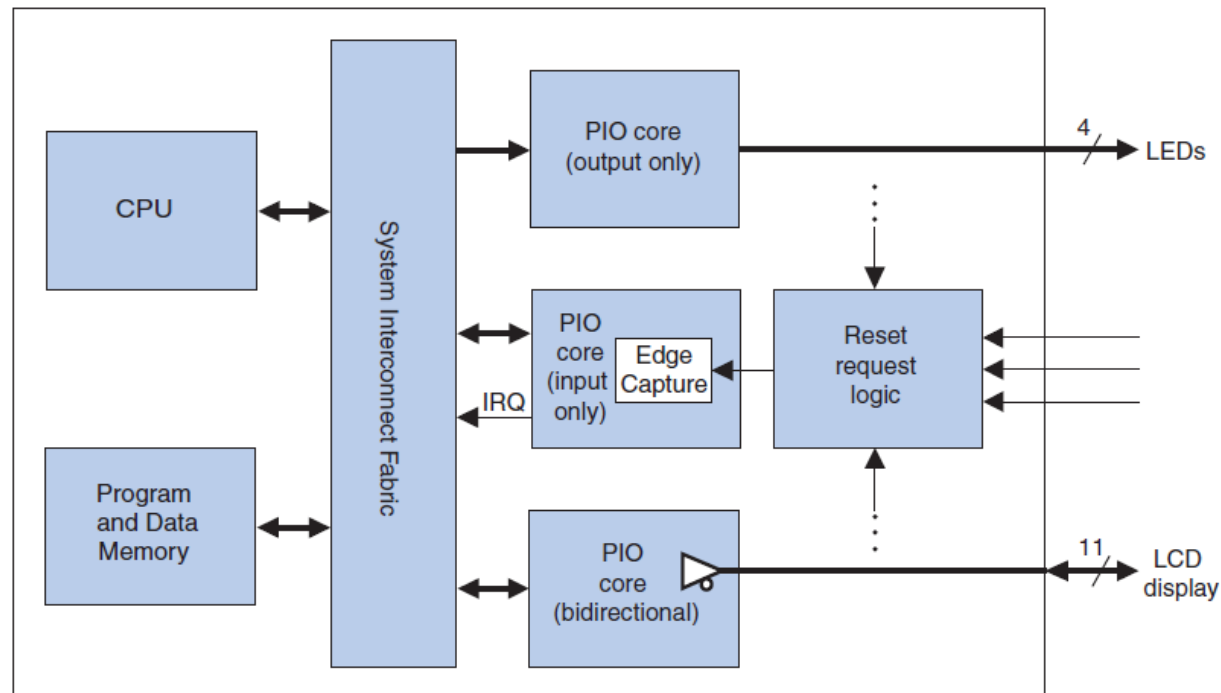
Parallel port (2)

- 4 32-bit memory-mapped registers
- *n* actual number of I/O pins

Table 2. Parallel Port register map

Offset in bytes	Register name		Read/Write	Bits $(n-1)\ldots 0$
0	`data`	Input	R	Data value currently on Parallel Port inputs.
		Output	W	New value to drive on Parallel Port outputs.
4	`direction`		R/W	Individual direction control for each I/O port. A value of 0 sets the direction to input; 1 sets the direction to output.
8	`interruptmask`		R/W	IRQ enable/disable for each input port. Setting a bit to 1 enables interrupts for the corresponding port.
12	`edgecapture`		R/W	Edge detection for each input port.

Parallel port (3)

- ## Managing PIO in C program:

  - Use of pointers to *unsigned int* <u>initialized</u> with PIO base memory address (we'll learn soon other ways!)

Red_LEDs_ptr   0x10000000

Red_LEDS
0x10000000

0x1000000C

Slider_switches_ptr   0x10000040

Slider_switches
0x10000040

0x1000004C

```
volatile unsigned int *red_LED_ptr = (unsigned int *) 0x10000000; //red LED address
volatile unsigned int *sliders_ptr = (unsigned int *) 0x10000040;
                                                //SW slider switch address
```

Parallel port (4)

- Why **volatile** attribute?
  - I/O registers may change even if the program does not modify them!
    - The peripheral hardware may modify their contents
  - **Volatile** tell the compiler do not make any optimization to the code involving an object declared with the **volatile** attribute

Parallel port (5)

- Reading/Writing I/O registers:

```
*red_LED_ptr = *Slider_switches_ptr;
```
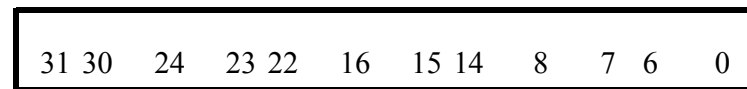
Putting into practice (1)

- Let's start our first program with Nios II processor
  - Control each DE2 red LED status through the corresponding slider switch (LEDR$_i$ = Sw$_i$)

Putting into practice (2)

- To go on:

  1. Show on the **4x 7-Seg `HEX3_HEX0` display** the
     4 <u>hexadecimal</u> digits of the 16-bit <u>unsigned</u> number
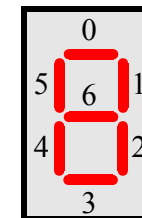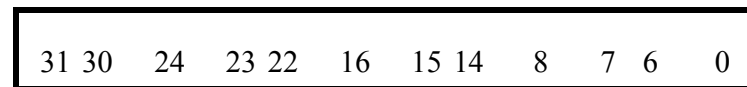   (Sw_{15}-Sw_0)

<u>Address</u>

0x10000020

| 31 30 | 24 | 23 22 | 16 | 15 14 | 8 | 7 6 | 0 |

Data register

HEX3$_{6-0}$ HEX2$_{6-0}$ HEX1$_{6-0}$ HEX0$_{6-0}$

```
    0
5   6   1
4       2
    3
```

Segments

0x10000030

| 31 30 | 24 | 23 22 | 16 | 15 14 | 8 | 7 6 | 0 |

Data register

HEX7$_{6-0}$ HEX6$_{6-0}$ HEX5$_{6-0}$ HEX4$_{6-0}$

Putting into practice (3)

- To go on:

  2. Show on the **4x 7-Seg HEX3_HEX0 display** the 4 <u>decimal</u> digits of the 16-bit <u>unsigned</u> number (Sw_{15}-Sw_0) if the number can be represented; **E** otherwise

  3. Allow the user to choice the representation between hexadecimal and decimal by the slider Sw_{17}

Putting into practice (4)

- To go on:

  4. Show on the **4x 7-Seg HEX3_HEX0 display** the module of the 16-bit <u>signed</u> number (Sw_{15}-Sw_0) and on $LEDG_8$ the sign of the number ($LEDG_8$ is ON if and only if the number is negative). Show the module using hexadecimal and decimal digits as before

  5. Allow the user to choice if (Sw_{15}-Sw_0) code an <u>unsigned</u> or <u>signed</u> number by the slider Sw_{16}

  6. Combine all the features in a single program

References

- Altera "Basic Computer System for the Altera DE2 Board"
- Altera "Parallel Port for Altera DE-Series Boards"