CO 513 – Microprocessors and Microcontrollers

Module 2: ARMv7 DE1-SoC I/O Devices

Electronics Engineering Department
College of Engineering
Technological University of the Philippines - Manila

ARMv7 DE1-SoC

An **ARMv7 DE1-SoC** is an educational development platform built around an Intel Cyclone V **System-on-Chip (SoC)**.

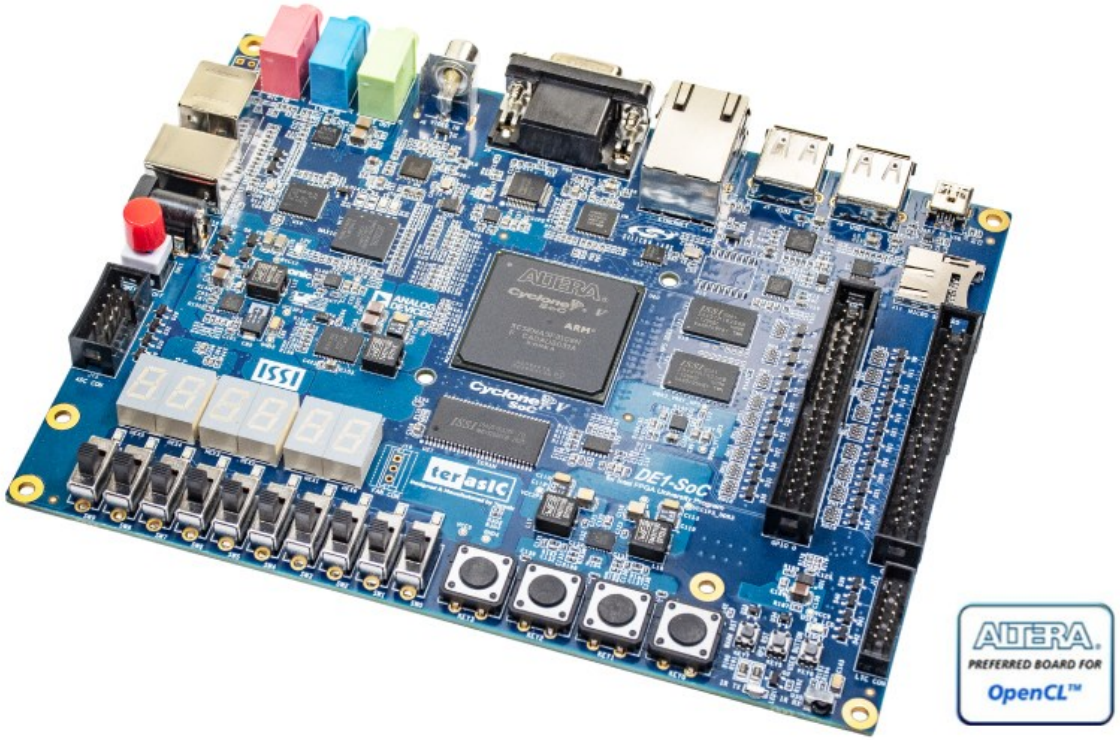

Image from: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=836

On CPUlator, the I/O devices are located at the right-hand side of the interface.

A complete list of peripherals are:

- Memory Regions
- Basic I/O Devices
- Communication Interfaces
- Timers
- Multimedia
- System Controllers

Memory Mapped I/O

Memory-Mapped I/O is a fundamental concept in embedded systems programming and computer architecture used to interact with hardware peripherals.

In systems that use Memory-Mapped I/O, hardware peripherals, such as the seven-segment display controller or switch inputs, are accessed by the central processor through specific addresses within the processor's standard address space

Memory Regions

DDR3 SDRAM (HPS) - `0x00000000–0x3FFFFFFF`

- 1GB main system memory

SDRAM - `0xC0000000–0xC3FFFFFF`

- 64MB general purpose RAM

SRAM (Pixel buffer) - `0xC8000000–0xC803FFFF`

- 256KB framebuffer for VGA graphics

SRAM (Character buffer) - `0xC9000000–0xC9001FFF`

- 8KB text mode VGA buffer

On-chip SRAM - `0xFFFF0000–0xFFFFFFFF`

- 64KB fast on-chip memory

Basic I/O Devices

LEDs - `0xFF200000–0xFF20000F`

- 10 red LEDs for output display

Switches - `0xFF200040–0xFF20004F`

- 10 slide switches for binary input

Push Buttons - `0xFF200050–0xFF20005F`

- 4 push buttons with interrupt capability

Seven-segment displays - `0xFF200020–0xFF20002F`

- (HEX3-0) - Rightmost 4 hex displays

Seven-segment displays - `0xFF200030–0xFF20003F`

- (HEX5-4) - Leftmost 2 hex displays

Communication Devices

PS/2 Port 1 - `0xFF200100–0xFF200107`

- PS/2 keyboard or mouse interface

PS/2 Port 2 - `0xFF200108–0xFF20010F`

- Second PS/2 keyboard or mouse interface

JTAG UART - `0xFF201000–0xFF201007`

- Serial communication for debugging/console I/O

Parallel Port 1 - `0xFF200060–0xFF20006F`

- General purpose parallel I/O

Parallel Port 2 - `0xFF200070–0xFF20007F`

- Second parallel I/O port

Carworld UART - `0xFF211020–0xFF211027`

- UART for car world simulation

Timers

Interval Timer 1 - `0xFF202000–0xFF20200F`

- Programmable timer with interrupt support

Interval Timer 2 - `0xFF202020–0xFF20202F`

- Second programmable timer

Cortex-A9 Private Timer - `0xFFFEC600–0xFFFEC60F`

- CPU private countdown timer

Cortex-A9 Watchdog Timer - `0xFFFEC620–0xFFFEC637`

- Watchdog timer for system monitoring

HPS L4 Watchdog Timer 1 - `0xFFD02000–0xFFD02017`

- Hard processor system watchdog

HPS L4 Watchdog Timer 2 - `0xFFD03000–0xFFD03017`

- Second HPS watchdog timer

Programming Demonstration 1

```
.global _start

/* Device base addresses */
.equ SWITCH_BASE, 0xff200040
.equ LED_BASE,    0xff200000

_start:
    /* Load base addresses */
    LDR r0, =SWITCH_BASE    // Switch base address
    LDR r1, =LED_BASE       // LED base address

main_loop:
    /* Read switch values */
    LDR r2, [r0]            // Load switch state into r2

    /* Write to LEDs */
    STR r2, [r1]            // Store switch state to LEDs

    /* Repeat forever */
    B main_loop

.end
```

m2_demo1.s

In this simple program, the LEDs reflect the states of the switches.

The base addresses of both devices are loaded into registers using LDR.

Then, the switch values are read indirectly through the address in r0 and stored in r2.

Finally, the same value is written to the LED address in r1, turning LEDs ON or OFF accordingly.

This demonstrates register indirect addressing using memory-mapped I/O.

Programming Demonstration 2

For the second programming demonstration, please see the m2_demo2.s file.

In this program:

- **SW0 ON** runs the scan to the right; **SW0 OFF** stops.
- New ON starts the LED at where it previously stopped.
- We poll the switch, use **AND/CMP** to test SW0, shift the one-hot bit (LSR) and **wrap** to LED9.
- Delay is a big busy-wait (LDR =2000000) for slow motion.

Programming Demonstration 3

For the third programming demonstration, please see the m3_demo2.s file.

The program continuously reads 8 switches (SW0-SW7) and displays their values as two hexadecimal digits on seven-segment displays:

- **SW0-SW3** (lower 4 bits) → **HEX0** (rightmost display)
- **SW4-SW7** (upper 4 bits) → **HEX1** (second display)

Thank You!

Module 2: ARMv7 DE1-SoC I/O Devices

Engr. Timothy M. Amado
Faculty, Electronics Engineering Department
Technological University of the Philippines – Manila