**RV32I Class Project Description**

1. Skeleton Analyzing
   1. RV32I System
      1. RV32I System.v
         1. Input: Clock / Button[3], Switch[10]
         2. Output: HEX0~3[7] / LEDG[10]
         3. Code
            1. Reset button = BTN[0]
            2. fetch addr, inst, data addr, write data, read data: Data fetch from MEM
            3. cs = chip select, data we = data write enable
      2. Altera Mem Dual Port: ram2port package / Inst/Data 포트
      3. Altera PLL: ALTPLL clkgen package / Clock기반으로 다양한 위상의 Clock 생성
      4. Decoder: Addr Decoder.v / 주소에 따라 cs 신호를 할당
      5. GPIO: GPIO.v / General Purpose I/O 모듈, 버튼/스위치/LED/7Seg 출력 관리
      6. RV32I CPU
         1. Basic Module.v: register I/O, ALU, Adder(32 bit, 1 bit), 2 input Mux
         2. rv32i cpu.v
            1. Controller

Maindec

Add ALU operations

Aludec

Add ALU signals for instructions on R/I-type instructions

* + - * 1. Datapath
    1. Timer: TimerCounter.v / 타이머/카운터 모듈
    2. RV32I System tb.v
  1. RV32I System SIM
  2. RV32I System SYN

1. Lab#1: Display Student ID to DE0 board
   1. Memory Map of Sevenseg

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

Write data to HEX0~HEX3: address +4

* 1. To display 1097 on SevenSeg
     1. 1 – (to 7seg)000\_0110 – (invert)111\_1001 - (to decimal) 121 – (to hex) 0x79 = (HEX3) 0x79
     2. 0 - (to 7seg)011\_1111 – (invert)100\_0000 - (to decimal/hex) 64/40 = (HEX2) 0x40
     3. 9 - (to 7seg)110\_1111 – (invert)001\_0000 - (to decimal/hex) 16/0x10 = (HEX1) 0x10
     4. 7 - (to 7seg)010\_0111 – (invert)101\_1000 - (to decimal) 88 – (to hex) 0x58= (HEX0) 0x58
  2. Code(C, Asm)

|  |  |
| --- | --- |
| 텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진  자동 생성된 설명 | 텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진  자동 생성된 설명 |

* 1. Expand: LED light on
     1. LED memory addr: 0xFFFF\_2008
     2. 10 bit exhibit each light
        1. 예시: 11\_1110\_0000 = 앞의 5개 light on
        2. Code
           1. **unsigned** **int**\* led = (**unsigned** **int**\*) 0xffff2008;
           2. \*led = 0x1e0; 0x1e0 = 11\_1110\_0000 in binary