교과목 명	전자 하드웨어 설계							
설계 제목	가스 벨브 안전 제어 시스템							
설계 기간	2017도 2학기							
지도교수	김태환							
팀원	**이름**	오혜빈	**학번**	2015124129	☎	010-2699-5260	**E-mail**	ohb0613@gmail.com
	이름	이민호	학번	20121222192	☎	010-2928-1686	E-mail	alsghdjrk@naver.com
	이름	임찬영	학번	2012122246	☎	010-3790-3347	E-mail	Limchan1@naver.com

목표설정

설계 목표

이 프로젝트의 목표는 가스 벨브를 안전 제어 하는 시스템을 설계하는 것이다. 세부적인 사항은 다음과 같이 정리했다.

1. C언어를 이용하여 가스나 화제의 발생시 사용자가 interrupt 기능을 수행한다. Interrupt handling을 통하여 4가지 (switch, senor, timer, push button) 기능을 control 할 수 있도록 한다.

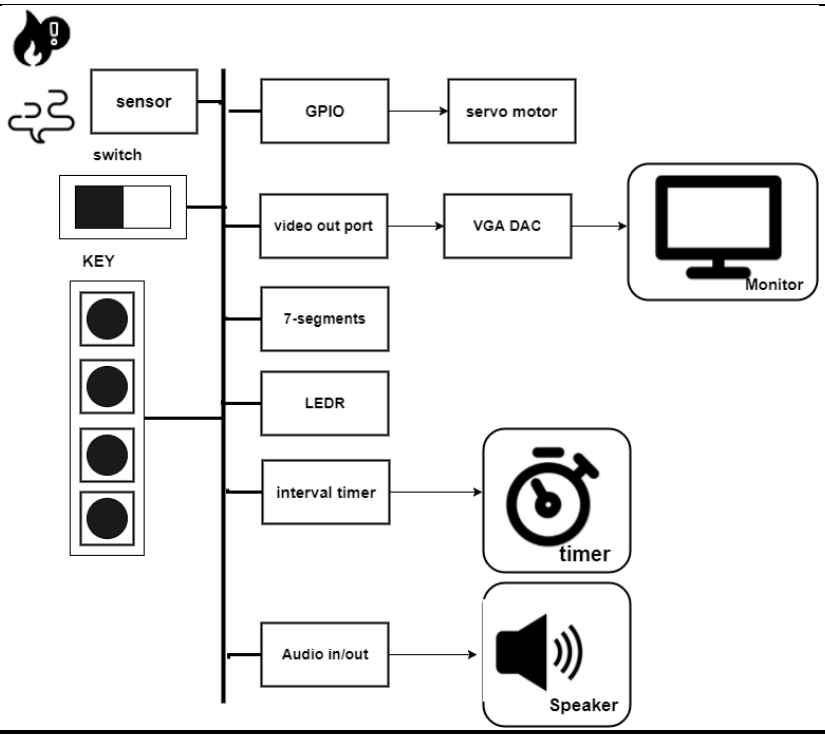
2. KEY를 이용하여 timer의 값을 받도록 한다. Reset, 타이머 시간증가, 타이머시간 감소, start/stop이 동작 하도록 구현한다.

3. sensor(MQ-2 센서와 같은 MQ-2 센서의 기능은 다음과 같다. {LPG, 부탄가스, 프로판가스, 메탄가스, 알콜, 수소가스, 연기 등을 감지})을 통하여 가스, 연기 등을 감지하게 되면 motor을 이용하여 가스 밸브를 잠글 수 있는 기능을 구현 할 것이다. Sensor를 통하여 interrupt가 발생하게 되면 motor를 작동시키게 구현 할 것이다.

4. sensor는 2개이상 구현할 것이다. GPIO를 이용하여 연결할 것이다. 2개 이상의 센서 중에 감지되는 sensor가 있으면 해당되는 sensor의 숫자혹은 위치(상하 또는 좌우)를 peripheral 중 video_out을 이용하여 화면에 display한다. 또한 LEDR을 이용하여 sensor에서 이상이(연기나, 가스등)감지 되면 LEDR에 이상이 있는 센서의 번호에 불을 밝힌다. 또한 센서에서 이상이 발생시 스피커를 이용하여 경고음이 나도록 한다.

5. 일정시간 시간이 지나면 servo motor을 이용하여 가스 밸브를 잠글 수 있는 기능을 사용한다. 시간을 사용하는 데에는 DE1-SOC 내부의 timer을 이용 할 것이며 작동 방법은 타이머로 지정한 시간이 지나면 자동으로 밸브가 자동으로 잠가지게 할 것이다. Timer을 이용한 시간을 7-SEGMENT에 display 할 것이며 timer의 시간이 다 지나게 되면 motor을 이용하여 밸브를 잠그게 구현할 것이다

설계 규격



Project에서 필요한 I/O port들을 위의 그림으로 정리하였다. 먼저 interrupt로 사용될 sensor와 push button, timer가 있고 그 외의 I/O port로 사용될 VGA DAC port와 7SEGMENT 그리고 motor와 센서의 감지를 표시할 LEDR, speaker가있다. 이 프로젝트에서 필요한 각각의 성능은 다음과 같다. 보드의 gpio에 센서를 연결하여 이상 경우(불꽃, 가스)발생 시 input을 받을 것이다. 이상 경우가 감지 되면 DE1SOC에 interrupt가 발생하게 하는데 사용된다. 이 interrupt가 발생하게 되면DE1SOC의 VGA DAC를 이용하여 monitor에 이상이 감지된 센서를 표시 하도록 하고 보조적으로 LEDR에 interrupt를 발생시킨 센서의 번호의 불을 키도록 한다. 뿐만 아니라 interrupt가 발생하게 되면 스피커가 울리게 하고 해당된 motor를 이용하여 밸브를 잠근다. Motor는 센서와 마찬가지로 gpio를 이용해 보드와 연결할 것이다. 각 상황별로 motor에 output값을 다르게 주어 벨브의 작동을 표현할 것이다. DE1SOC의 KEY를 이용하여 timer의 시간, on/off, reset을 관리 한다. Timer은 카운트 다운을 지속적으로 7SEGMENT에 남은 시간을 표시하기위 또한 시간이 0가 되면 motor을 작동시키기 위해 사용하게 된다. 다음으로 timer는 100Mhz를 이용하여 1second를 기준으로 만들어 사용한다.

합성/분석

관련 기술

인터럽트 기반으로 De1-soc 보드의 기능들을 사용하기 위해 I/O Peripheral의 값을 Ienable 부분에 Write 해주는 ctl3 register, STATUS에 인터럽트를 받을 수 있는 PIE의 값을 1로 초기화 해준다. 이 프로젝트에서는 Interval timer, Pushbutton, JP1 Expansion parallel port의 interrupt를 사용할 것으로 예상된다.

Register	Name	b_{31} ⋯ b_2	b_1	b_0
ctl0	status	Reserved	U	PIE
ctl1	estatus	Reserved	EU	EPIE
ctl2	bstatus	Reserved	BU	BPIE
ctl3	ienable	Interrupt-enable bits		
ctl4	ipending	Pending-interrupt bits		
ctl5	cpuid	Unique processor identifier		

I/O Peripheral	IRQ #
Interval timer	0
Pushbutton switch parallel port	1
Second Interval timer	2
Audio port	6
PS/2 port	7
JTAG port	8
Serial port	10
JP1 Expansion parallel port	11
JP2 Expansion parallel port	12

-Memory mapped I/O

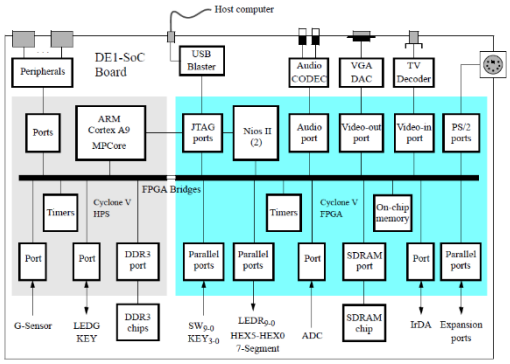


Figure 1. Block diagram of the DE1-SoC Computer.

입출력 장치를 제어하기 위해 입출력 장치의 레지스터에 메모리와 같이 주소를 할당하고, 데이터를 읽고 쓴다. Memory mapped I/O 프로젝트를 진행하면서 우리는 NIOS2에서 기본으로 제공하는 address.map_nios2.h를 이용하여 memory mapped I/O를 진행한다. 이 프로젝트에서는 VGA-DAC, parallel ports(expansion ports, SW, KEY, HEX, LEDR),Timers를 사용할 것이다.

-Expansion parallel port

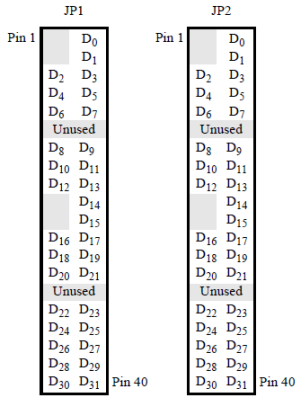


Figure 7. Assignment of parallel port bits to pins on *JP1* and *JP2*.

DE1SOC 보드에는 2개의 40 pin으로 구성된 JP1, JP2 parallel ports가 있다. 이 parallel port는 32bit register(Data register, Direction register, Interruptmask register, Edgecapture register)를 포함한다. 보드의 GPIO와 센서, 모터를 연결하여 제어할 것이다.

- Interval Timer

Address	31 ... 17	16	15 ... 3	2	1	0	
0xFF202000	Not present (interval timer has 16-bit registers)		Unused		RUN	TO	Status register
0xFF202004			Unused	STOP	START	CONT	ITO Control register
0xFF202008			Counter start value (low)				
0xFF20200C			Counter start value (high)				
0xFF202010			Counter snapshot (low)				
0xFF202014			Counter snapshot (high)				

이번 프로젝트에서 사용할 Interval Timer는 인터럽트를 기반으로 한 Timer이다. 현재 타이머의 시간을 7segment에 나타내 주면서 알람, 타이머 등 여러 기능을 인터럽트로 발생시킬 것이며 그림을 보면 status register의 두번째 bit RUN은 TIMER가 동작하기 위해 1로 설정해야 하며 TO는 1초가 count되면 TO가 0에서 1로 바뀐다. 다음 1초를 count하기 위해 TO를 다시 0으로 초기화 해준다. Control register는 STOP, START, CONT, ITO가 있다. STOP은 현재 Interval Timer의 동작을 멈추게 하고 START는 타이머가 동작을 시작함을 나타내고 CONT는 타이머가 계속 동작하게 해준다. ITO는 Interrupt Time Out으로 인터럽트가 발생하면 ITO의 값이 1이 된다 다시 인터럽트로 Interval TIMER를 사용하려면 값을 0110으로 초기화 해주면 된다. 그 다음의 레지스터에는 value(주파수)를 write하여 Timer의 시간이 몇 초 주기로 time out될지 설정해준다.

-Pushbutton switch parallel port

Address	31 30 ... 4 3 2 1 0	
Base	Input or output data bits	Data register
Base + 4	Direction bits	Direction register
Base + 8	Mask bits	Interruptmask register
Base + C	Edge bits	Edgecapture register

4개의 Pushbutton을 사용하기 위해 0b1111을 Interruptmask register에 load해준다. 각 버튼의 입력 상태는 Edgecapture register를 읽어서 알 수 있다. 0b0001, 0b0010, 0b0100, 0b1000과 값을 비교하여 어떤 key가 눌렸는지 판단한 후 각각의 버튼에 맞는 기능을 수행하는 service code로 들어 가게 된다.

- Audio IN/OUT

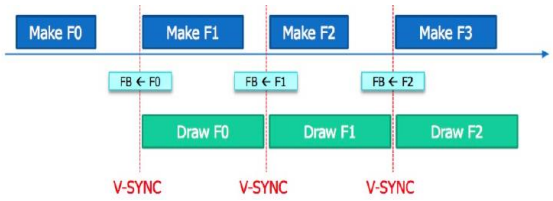
Address	31 ... 24	23 ... 16	15 ... 10	9	8	7 ... 3	2	1	0	
0xFF203040	Unused			WI	RI		CW	CR	WE	RE Control
0xFF203044	WSLC	WSRC	RALC			RARC				Fifospace
0xFF203048	Left data									Leftdata
0xFF20303C	Right data									Rightdata

Figure 25. Audio port registers.

Audio port는 32비트 레지스터로 구성 되어 있으며 RE부분은 입력된 data를 위한 인터럽트 enable을 제공한다. RI bit가 1이면 인터럽트가 pending된 것이다. WE bit는 outgoing data를 위한 인터럽트 enable해주는 부분이다.

-Video Output(Double buffering)

움직이는 이미지를 출력할 때, 하나의 버퍼를 동시에 읽고 쓰면 화면이 깜박이거나 깨지는 문제가 발생한다. 이를 해결해주기 위해 back buffer와 front buffer를 parallel하게 이용하는 double buffering 방법을 이용한다. Backbuffer에 그려진 후 S bit가 0이 되는 것을 기다린 후 이를 front buffer에 옮겨주고, S bit가 1로 바뀐다. 이후 다시 v-sync를 맞추어 화면에 출력해 준다.

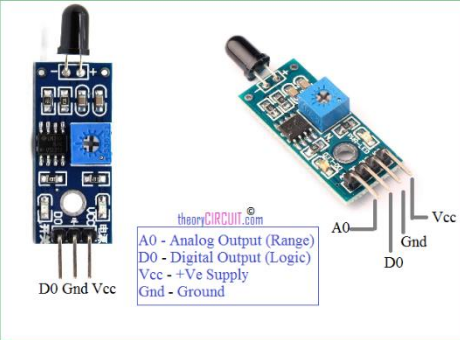


-MQ-2 Gas sensor

가스를 감지하기 위해 MQ-2 sensor을 사용할 가능성이 가장 높다. MQ-2는 LPG, 부탄, 메탄, 알코올을 검출 할 수 있다. 센서 내부에 있는 히터가 가열이 되면 센서 내부의 금속막에 공기중의 성분이 달라붙게 된다. 이때, 금속막에 성분이 달라붙음에 따라 저항 값이 달라져서 가스를 감지할 수 있다. 가스를 감지하면 interrupt를 발생시키도록 할 것이다. 위의 사진을 보면 핀이 4개가 있다. vcc에는 전원을 연결해주고, gnd에는 ground를 연결해준다. 그리고 AO 핀에서는 센서로 감지된 값이 아날로

그로 출력되고, DO 핀에서는 센서로 감지된 값이 디지털로 출력된다. 가변저항으로 감도를 지정해주고 DO 핀과 DE1SOC보드의 gpio핀을 연결해주어 센서의 출력 값을 받을 수 있다.
< https://www.amazon.com/Wavesahre-MQ-2-Gas-Sensor-Detection/dp/B00NJOIB50>

-HS-FLAME sensor

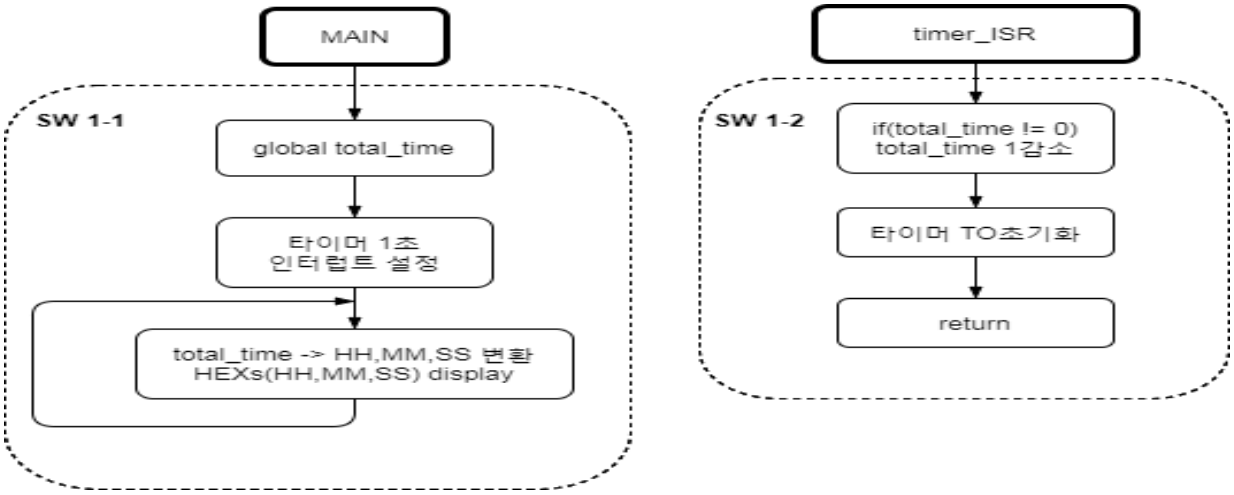


스파크와 같은 불꽃을 감지하기 위해 flame sensor를 사용할 것이다. 이 센서는 760nm~1100nm의 파장을 갖는 빛, 불꽃을 감지할 수 있다. 감지 각도는 60도 이다. 센서에 있는 가변저항으로 감도를 조절할 수 있다. 핀은 4개가 있다. 이는 위의 MQ-2 센서와 같다. DO 핀과 DE1SOC보드의 gpio핀을 연결해주어 센서의 출력 값을 받을 것이다.
< http://www.theorycircuit.com/arduino-flame-sensor-interface/>

<위의 레지스터에 관한 그림들은 DE1SOC_computer_Nios2.pdf에서 참조했다.>

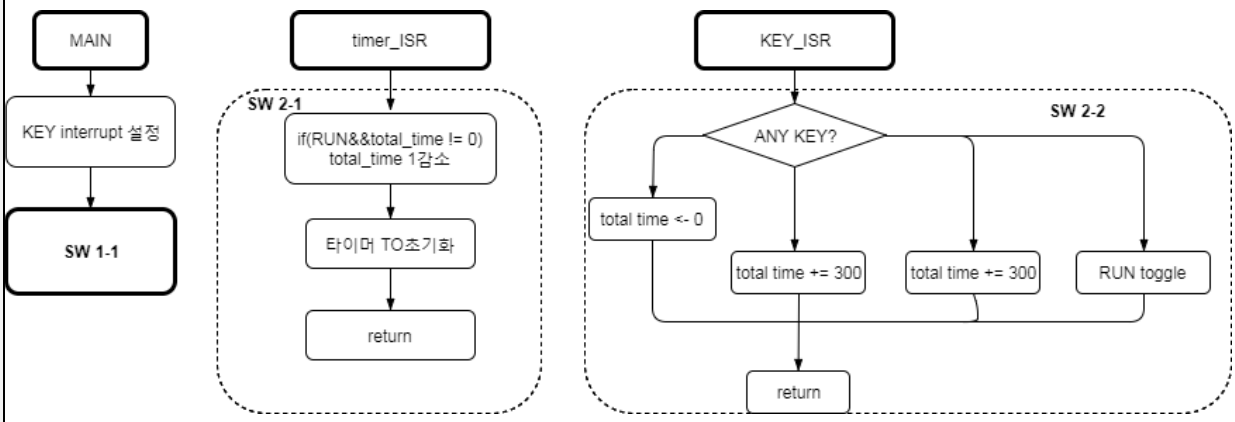
제작 설계 계획

1) SW1 - 기본 timer를 이용한 시간 구현



SW1-1은 MAIN함수의 global 변수 total_time을 받아서 timer자체는 1초마다 인터럽트를 일으켜서 timer_isr를 호출한다 이 timer_isr의 기능은 1초마다 global total_time을 1씩 감소시키는 역할을 하고 이 감소 시킨 숫자를 return하여 SW1-1 에서 7SEGMENT로 변경하여 HEX에 출력하는 동작을 담당 하는 것이다. 이 과정을 0이 될 때까지 반복을 한다.

2) SW2 - key를 이용한 timer의 변수 global_time의 값 조정

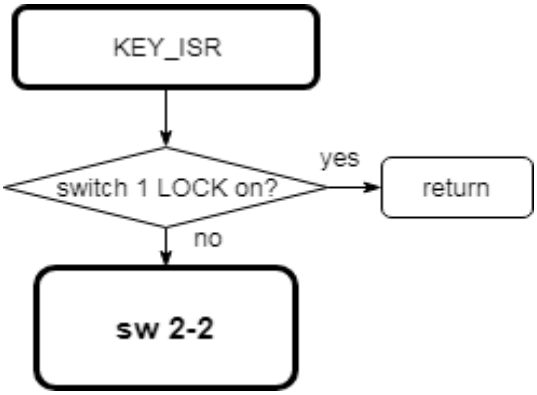


DE1SOC의 KEY_BASE의 4개의 키를 사용하기 위해서 interrupt mask bit에 0b1111을 저장 하고 시작할 것이다.

각 키의 버튼이 눌리고 때 졌을 때 함수를 발생 시키도록 한다. 첫번째 키가 눌렸다가 때질 때 시간을 초기화 시킨다.(초기값은 1시간으로 할 예정이다) 두번째 키는 시간 추가(5분), 세번째 키는 시간감소(5분), 4번째 키는 global 변수 timer run을 toggle한다.

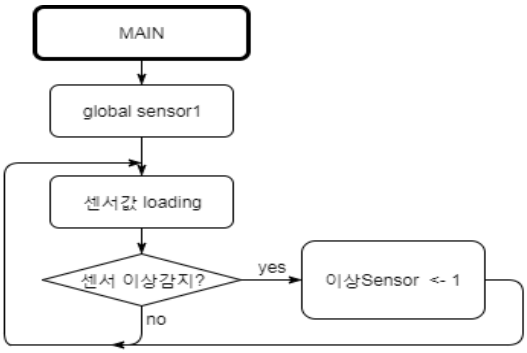
3)SW-3 스위치를 이용한 KEY LOCK기능

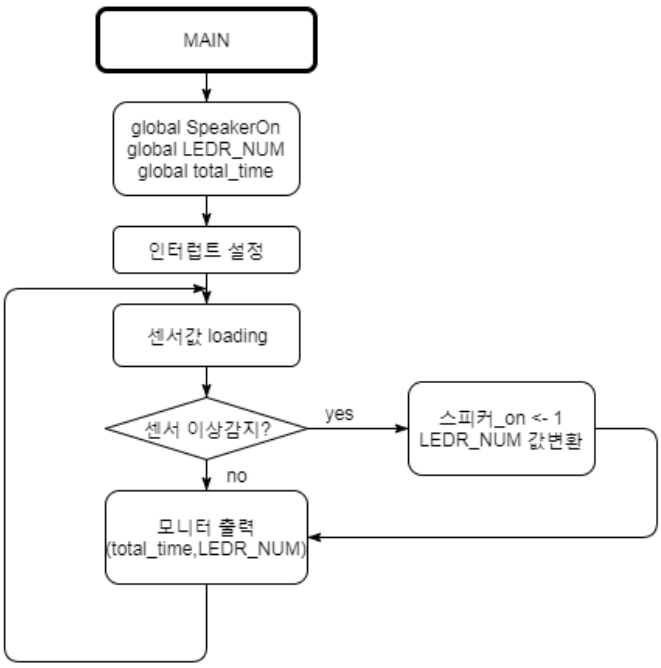
가스감지기의 컨트롤함에 있어 실수로 버튼이 눌려 오작동하는 것을 막기 위해 NIOS-2에 내장되어 있는 switch활용하여 KEY LOCK기능을 구현하려 한다. 원리는 다음과 같다. KEY가 눌려 interrupt가 걸리고 KEY_ISR이 실행됐을 때 switch값을 읽어 들이고 switch 1에 해당하는 비트를 확인하게 된다. Switch 1 bit값이 없다면 SW2-2(key를 이용한 스탑워치 controll()실행되어 Service를 하고 종료하게 되고 bit값이 있다면 KEY LOCK이 걸려있고 SW2-2이 실행되지 않고 key_ISR을 종료하게 된다.

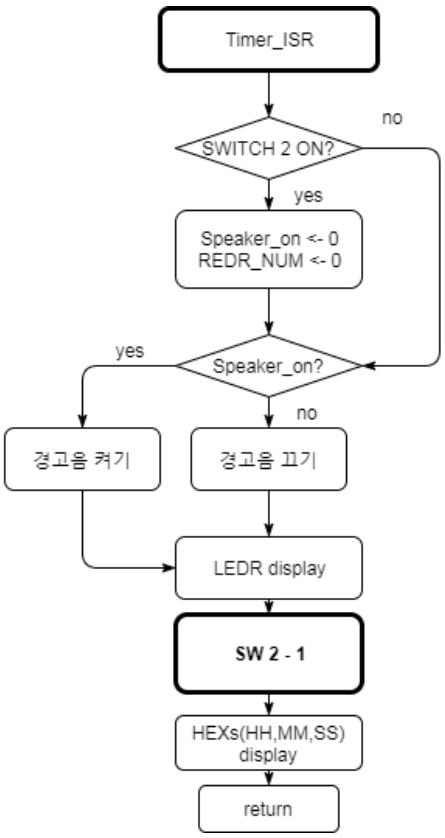


4) SW - 4 센서 값 읽기

가스감지기의 주요역할 중 하나는 가스를 감지하여 대처를 하여 사고를 방지함에 있다. 이 시스템에서는 Polled IO방식으로 반복문을 돌며 Sensor(MQ-2)의 input값을 loading하여 읽게 된다. 후에 센서 이상이 감지됐을 경우에 global sensor1의 값을 1로 바꾸게 된다. 후에 멀티 센서를 뒀을 때 여러 sensor변수를 해당 센서에 값을 줄 예정이며 이 글로벌 변수를 통하여 해당 LEDR표시, 모니터출력, 스피커 경고음, 모터 잠그기 등 제어에 이용되도록 설계할 예정이다.







5-1) LEDR 감지

센서가 가스를 감지시 전역변수 해당 SENSER에 값이 바뀌고 LEDR_NUM값에 해당센서 비트에 값을 넣어준다.

		timer_interrupt가 1초마다 발생하게 될 때 LEDR에 LEDR_NUM 표시한다. 스위치 2번째를 키면 timer_ISR에 스위치에 2번 bit를 확인하여 전역변수 LEDR_NUM의 값을 0으로 바꿔주게 되고 경고음이 꺼지게 된다. 5-2) 모니터 출력 모니터에 스탑워치 시간, 이상 센서값을 모니터에 출력한다. 5-3) 스피커 경고음 센서가 가스를 감지시 스피커로 통하여 경고음을 발생시켜야 한다. 동작원리는 다음과 같다. SW-4에서 가스감지시 speakerON변수값이 1이 된다. 그리고 timer_interrupt가 1초마다 발생하게 될 때 speaker_on값을 확인하여 스피커를 끄고 켜고 하게된다. User가 경고음을 듣고 경고음을 끄기 위해서는 스위치 2번째를 키면 timer_ISR에 스위치에 2번 bit를 확인하여 전역변수 Speaker_on의 값을 0으로 바꿔주게 되고 경고음이 꺼지게 된다. 5-4) 서보 모터 제어 센서가 가스를 감지시 안전을 위해 가스를 모터를 이용하여 잠그려 한다. 동작원리는 다음과 같다. SW-4에서 가스를 감지시 전역변수 motor_time값을 잠기는데 걸리는 시간(예상 5초)로 설정되게 된다. 그리고 timer_interrupt가 1초마다 발생하게 될 때 motor_time값이 0이 아니라면 모터를 잠그도록 회전시키고 1만큼 감소시킨다. 후에 0되었을 때 가스가 모두 잠기며 모터가 정지하게 된다. 모터를 여는 것은 switch 3번으로 모터를 열 예정이고 모터를 잠겼다는 것을 변수로 표시하려고 한다.
시험/평가	검증계획	1순위로 interval timer의 interrupt를 이용한 타이머가 제대로 동작하는지를 검증하고, 이를 모니터, 7segments 상에 display 할 수 있는지 확인한다. 그리고 다른 하드웨어(센서, 서보모터 스피커)들이 보드와 연결되어 제대로 작동하는지(input값을 받고 그에 따라 output값을 줄 수 있는지)를 각각 확인하고, 마지막으로 전체적인 가스 벨브 안전 제어의 알고리즘에 맞게 제대로 동작하는지를 확인한다. 1주차 타이머가 잘 작동하는지 검증해본다. 첫째로, KEY버튼의 동작에 따라 타이머가 작동하는지 검증한다. KEY0을 누르면 타이머의 시간이 reset되고, KEY1를 누르면 timer의 시간을 5분 추가하고,, KEY2를 누르면 timer의 시간을 5분 감소, KEY3을 누르면 타이머의 동작을 시작/정지 되는지 확인해본다. Fpga monitor program의 터미널에 타이머의 시간을 출력하는 코드 printf를 이용하여 타이머가 제대로 작동하는지 확인해본다. 이 때 타이머가 제대로 작동하지 않는다면 pushbutton과 interval timer의 인터럽트가 발생하는지 알아보고, 타이머의 register의 RUN, TO, CONT, STOP bit도 terminal에 출력하여 확인해본다. 둘째로 타이머의 시간이 hex에 display되는지를 검증해야 한다 만약 HEX에 제대로 display 되지 않는다면 코드의 display 부분을 살펴보고 7segment형식으로 데이터가 제대로 변환되었는지 계산해본다. 셋째로, switch로 타이머의 KEY가 lock되는 기능을 검증해야한다. Switch의 값이 1인상태에서 key0~1을 눌렀을 때 타이머가 제어되지 않는지 확인해본다. 만약 lock되지 않는다면 switch의 인터럽트가 일어났는지를 확인해본다. 2주차 첫번째로 타이머의 상태를 케릭터 버퍼를 이용해 모니터에 출력 해야한다. 1주차에 타이머의 동작을 완료했으므로 모니터에 출력이 되는지만 검증하면 된다. 모니터에 출력되었을 때 화면의 깜박임 등 문제가 없는지도 체크한다. 두번째로, gpio에 연결된 센서로부터 그 출력값을 받을 수 있는지 확인해야한다. 먼저 센서의 출력값을 임시의 변수에 저장하여 terminal상에 출력하여 확인해본다. 그 후 센서의 출력값이 1일 경우 각 센서에 해당하는 LEDR이 ON되는지 확인해본다. 센서의 출력값이 1이되도록 해주기 위해서 센서 근처에 라이터를 가져다 대본다. 센서 자체의 문제일 수 있으니 여러 개의 센서로 테스트해본다.

3주차
첫번째로, gpio를 통해 서보모터가 제대로 연결되었는지 확인해야한다. 서보모터만 따로 작동하는지 확인하기 위해 서보모터를 90도, 180도 움직이는 코드를 작성하고 실제로 작동되는지를 확인해본다.
둘째로, 센서의 값에 따라 서보모터가 작동하는지 확인해보아야 한다. 위의 동작까지 모두 검증이 된 상태에서 이 단계가 제대로 동작하지 않는다면 이는,하드웨어 연결의 문제가 아니고 알고리즘적인 문제이다. 센서값과, 모터의 입력값을 terminal 상에 출력해보고 이것이 작성한 코드의 알고리즘에 맞는지 확인해보고 코드를 수정해본다.

둘째로, 타이머의 시간이 끝나면 서보모터를 잠궈야 한다. 위와 같은 방법으로 검증해본다.
셋째로, 샌서의 동작이 모니터에 표시되는지 확인해 봐야한다.

넷째로, 센서에 감지가 되거나 타이머의 시간이 다되면 경고음이 출력되어야 한다. 알람음이 audio out port로 출력이 되는지 확인한다.

일정

주차	진행 계획	예상 산출물 및 Demo 내용
1	11월 24일~11월 30일 -주제 설정, 회의 -설계 계획서 작성 -전체적 일정 및 설계목표 단계 결정	12월 1일 -설계계획서 제출
2	12월 1일~12월 7일 - 타이머 기능(hex에 display) - Key 동작 구현(리셋, 시간증가, 시간감소 , on/off) -switch를 이용해 lock 기능 -LEDR에 표시 -사용할 센서, 부품 결정, 구매 -회의록 1 작성	12월 8일 타이머 전체적인 동작 확인 -> key에 따른 세부사항 동작 ->동작(KEY, Switch)에 따라 Hex, LEDR display ->switch를 on했을 때 key를 눌러도 timer의 동작에 변화가 없는지 확인 (lock) -회의록1 제출
3	12월 8일~12월 14일 - gpio사용법을 공부하고, 이를 이용하여 sensor 연결 - 케릭터 버퍼를 이용하여 모니터에 시간과, 센서의 감지여부를 출력한다.(어떤 센서에 감지 됐는지) -회의록2 작성	12월 15일 모니터에 타이머상태를 출력 ->모니터에 타이머의 시간을 출력 ->모니터에 센서의 감지여부를 출력 ->화면 깨짐, 깜박거림이 없는지 확인 - 회의록2 제출
4	기말고사 기간	x
5	12월22일~12월 25 - 서보모터를 보드에 연결한다. - 센서의 값에 따른 서보모터를 이용	12월 26일 -결과보고서 제출 -최종 작동 시연

		한 동작 실행. -타이머 값에 따른 서보모터를 이용한 동작 실행. - 센서의 동작 표시 - 경고음 출력 - 결과보고서 작성 - 공학인증 포트폴리오 업로드 - 최종시연 준비	->서보모터의 동작 확인 ->타이머 값, 센서의 감지여부에 따른 서보모터의 동작 확인 ->경고음 출력

역할 분담	<오혜빈> 1. gpio를 이용해 보드에 센서를 연결. -센서를 연결하여 인풋 값 받기위한 함수 구현 2.서보모터를 보드의 gpio에 연결 -서보모터를 제어하기 위한 함수 구현 3. 회의록 작성 4. 결과보고서 작성 <이민호> 1. 모니터에 화면을 출력하기 위한 code 작성 -캐릭터 버퍼를 이용하여 타이머 시간표시 -센서의 감지여부 표시 -모터의 동작여부 표시 2. 센서의 감지여부, 타이머의 타임아웃에 따라 경고음을 출력하기 위한 함수 구현 3. 회의록 작성 4. 결과 보고서 작성 <임찬영> 1. TIMER 구현 -key , switch에 따라 start/stop, lock, 시간설정 동작이 되도록 함 -key0: reset, key1: 시간 증가, key2: 시간 감소, key3: start/stop, sw : key lock -타이머의 시간을 hex에 표시하기 위한 함수 작성 2.timer의 타임아웃인 경우와 센서 감지 시 서보모터를 제어하는 알고리즘 설계 3. 회의록 작성 4. 결과보고서 작성