```
//Modules: (go into each module and find these):
//Registers and MUX … which will go into the DP

//Counters (Control Unit)
//Clock divider (Control Unit)
//But the registers they use are DP
```

If sub is = 0 and Q is 0 then you were in adjust and want to go to clock
If sub is = 0 and Q is 1 then you were in clock mode and want to go to adjust

```
CurrentTime; (14 bit register)

3-0 minute one //Mod 10 so we need to have corresponding counters
6-4 minute tens //Mod 6 so we need to have corresponding counters
10-7 hour one //Mod 10 so we need to have corresponding counters
13-11 hour tens  //Mod 6 so we need to have corresponding counters

buttonState; (5 bit register)
0 BTNC
1 BTNL
2 BTNR
3 BTNU
4 BTND

SetAlarmTime; (14 bit register)

3-0 minute one
6-4 minute tens
10-7 hour one
13-11 hour tens



LD0 = 0;  //Led 0 (1 bit variable)
Dp2 = 0;  //Decimal point 2 (1 bit variable) //This is what makes it blink (this is a second)
ClockAdjust = 1; //Current mode (1 bit variable)
AlarmSound = 0;  //Should the alarm make a sound (1 bit variable)

if(buttonState_0 == 1) do
ClockAdjust = 0;
End if;

while(ClockAdjust  == 1) do
ENABLE CLOCKCounter;
if(buttonState_0 == 1) do
ClockAdjust = 0;
End if;

Dp2 = 1; //We should blink the decimal point (MUXs 0 and 1 as inputs and selection from control unit)
//Time registers = signal from the counter
if(CurrentTime == SetAlarmTime) //Zflag
AlarmSound = 1;
LD0 = 1; //This should blink //c2 this is what will make it blink
    while(buttonState == 0) do //No ones //We can send it to the CU so it can loop
        AlarmSound = 1;
        LD0 = 1;
    End while;
    AlarmSound = 0;
    LD0 = 0;
End if;
End if;

End while;
```

Should we stop the clock?

```
LD0 = 1;  //Led 0 (1 bit variable)
Dp2 = 0;  //Decimal point 2 (1 bit variable)
buttonState; (5 bit register)
LEDstate; (5 bit register)
parameter; (7 bit register)
//0 LD0
//1 LD12
//2 LD13
//3 LD14
//4 LD15
ClockAdjust = 0; //Current mode (1 bit variable) (0 means adjust mode)
CurrentTimeMin; (7 bit register)

CurrentTimeHour; (7 bit register)
SetAlarmMin;
SetAlarmHour;
//While in the time options we will display clock on 7SEG, while in the alarm options we will display alarm on 7SEG.
LEDstate_1 = 1;


if(buttonState_0 == 1) do
ClockAdjust = 1;
End if;

//Up/Down mod-4 counter for selection of 4x1 MUX that picks the bits that we want to change
//With buttonState_1 as -1 and buttonState_2 as +1
case(sel) //c1
0: CurrentTimeMin & LD12 = 1 & LD13 = LD14 = LD 15 = 0
1: CurrentTimeHour & LD13 = 1 & LD12 = LD14 = LD 15 = 0
2: SetAlarmMin & LD14 = 1 &  LD13 = LD12 = LD 15 = 0
3: SetAlarmHour & LD15 = 1 & LD12 = LD13 = LD 15 = 0
endcase

//Adjust
if(buttonState_3 == 1) do
TheOneSelected<= TheOneSelected+ 1;
end if;
if(buttonState_4 == 1) do
TheOneSelected<= TheOneSelected- 1;
end if;
```

Frequency in = 100Mhz = 1x10^8

We want one tick to be 60hz since 1hz is one second
and one minute is 60 seconds.

Frequency out = 60hz

60 = x*(1*10^8) = (6*10^-9)(1*10^8)