

Jonathan W. Valvano

March 26, 1999, 11:00 am to 12:50pm

(50) Question 1. Design a position DAS using a sensor, analog electronics and a 6812.

(5) Part a) $y=0.1V/mm \cdot x$. The sensitivity, which is the slope of the $y=f(x)$ response, is $0.1V/mm$.

(5) Part b) We wish to map $y=-0.1V$ into $z1=0$ and $y=+0.1V$ into $z1=+5$. The transfer equation is

$$z1=25 \cdot (y+0.1) = 25 \cdot y + 2.5$$

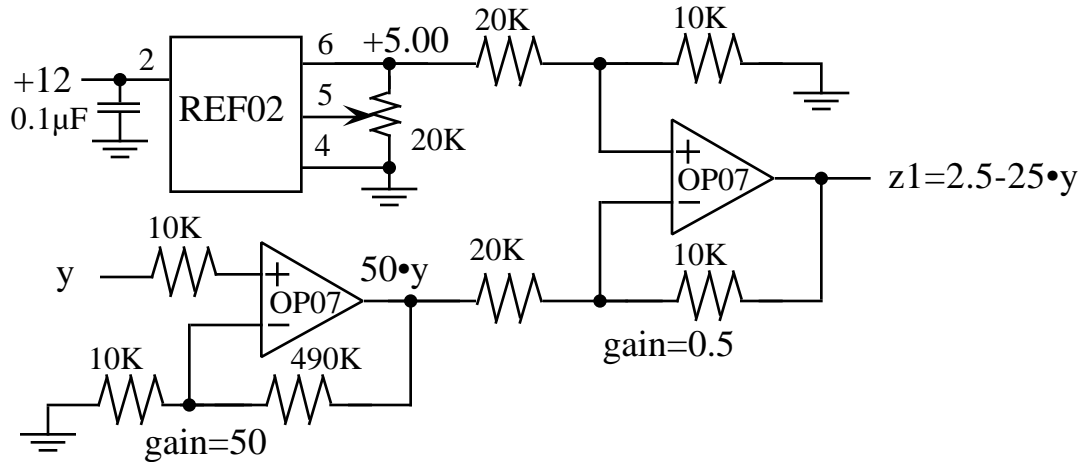
We could also map $y=-0.1V$ into $z1=+5$ and $y=+0.1V$ into $z1=0$. This transfer equation is

$$z1 = 2.5 - 25 \cdot y$$

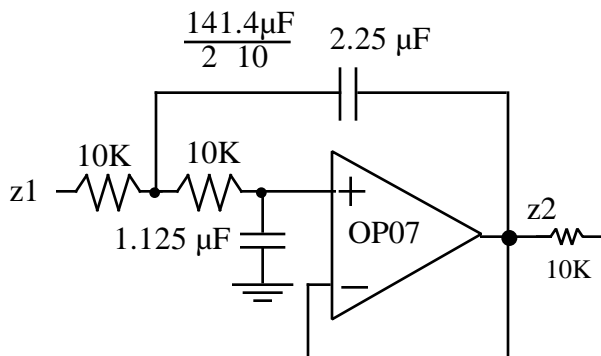
(5) Part c) We want the maximum allowable noise for our analog circuit to be less than the resolution at that point. Assuming the A/D is the limiting factor, the resolution at y is $0.2V/256$ or $781 \mu V$. Adding a safety factor of $1/2$, we want the amplifier noise to be less than $390 \mu V$ referred to its input.

(5) Part d) Nyquist Theorem states any sampling rate strictly greater than $2 Hz$ would be appropriate.

(15) Part e) We can implement the transfer relation, $z1 = 2.5 - 25 \cdot y$, with just two op amps following the example in Figure 11.41. Because the first stage is a noninverting amplifier the input impedance of the circuit will be large. The op amps will need to be powered with $+12$ and -12 volts because the intermediate signal, $50 \cdot y$, can range from -5 to $+5$ volts. The bandwidth of the first stage is $8KHz$ and the second stage $400KHz$, so the combined bandwidth is about $8KHz$, well above the specifications. All resistors are 1% metal film.



(10) Part f) You could scale the capacitors to standard values 0.1 and $0.2\mu F$ by changing the $10K$ resistors to $112.5K$. Since the exact cutoff of $10 Hz$ is not critical, you could use a $100K$ resistor making the cutoff about $11Hz$. All resistors are 1% metal film, and capacitors are polystyrene, polypropylene or Teflon.



(5) Part g) Assuming the A/D is the limiting factor, the system resolution will be $2/256=0.0078 mm$.

(50) Question 2. The objective of this problem is to design an underwater ultrasonic ranging system.

(5) Part a) The equation is $d = c \cdot t / 2$ or $d = 750m/sec \cdot t$

(5) Part b) At $d=1m$, $t=1.3 ms$. At $d=100m$, $t=133 ms$.

(5) Part c) At a time resolution of $4\mu\text{s}$, the distance resolution will be $750\text{m/sec} \cdot 4\mu\text{s} = 0.003\text{m}$. At this time resolution the measurement range does include the 1.3ms to 133ms time range. If t is measured in $4\mu\text{s}$ clock counts, then $d = 0.003\text{m/count} \cdot t$. If t is measured in $4\mu\text{s}$ clock counts and d is measured in 0.01m units, then the fixed point calculation will be $d = 3 \cdot t / 10$.

(10) Part d) Give the ritual which initializes the interface.

```
// PT1/OC1 output = generate sonic pulse via the transmitter
// PT1 output is a 5us pulse every 1 sec
// PT0/IC0 input = echo pulse from receiver
// PT0 interrupt on rising edge, resolution = 4us, Range = 1.3ms to 133 ms,
// IC0 interrupt on the first echo after an output pulse,
unsigned int Depth; // units of 0.01 m
unsigned int First; // TCNT at the start of the pulse output
unsigned int Time; // incremented every 250ms
void Ritual(void){
    asm(" sei"); // make atomic
    TIOS = 0x02; // PT1 is output compare, PT0 is input capture
    DDRT = 0x02; // PT1 is output, and PT0 is input
    TSCR = 0x80; // enable TCNT
    TMSK2= 0x35; // 4us clock
    TCTL4 = 0x01; // rising edge of IC0 sets COF flag
    InitFifo(); // will be filled about once a second
    TFLG1 = 0x02; // Clear C1F
    TMSK1 = 0x02; // Arm OC1, disarm IC0
    Time = 3; // first OC1 interrupt will send a pulse
    TC1=TCNT+62500; // First one in 250 ms
asm(" cli"); }
```

(5) Part e) The distance calculation is performed in the main program because it makes the interrupt handlers shorter.

```
void main(void){ unsigned short TimeOfFlight; long lDepth;
    Ritual();
    while(1){
        while(GetFifo(&TimeOfFlight)){}; // ranges from 333 33333 counts
        lDepth= (3*(long)TimeOfFlight)/10; // promote to 32 bits, calculate and
        Depth=lDepth; // demote back to 16 bits
        if(Depth<1500) Alarm(); // alarm if distance less than 15 m
    }
}
```

(10) Part f) The handler is executed every 250 ms, every 1 second a $5\mu\text{s}$ pulse is generated.

```
#pragma interrupt_handler TC1handler()
void TC1handler(void){
    TFLG1=0x02; // ack C1F
    TC1=TC1+62500; // Executed every 250 ms
    Time++;
    If(Time==4){ // Executed every 1 sec
        Time=0;
        PORTT=0x02; // start pulse
        First=TCNT; // begin time measurement
        PORTT=0x02; PORTT=0x02; PORTT=0x02; PORTT=0x02; PORTT=0x02; PORTT=0x02; // wait 5us
        TFLG1=0x01; // Clear COF
        TMSK1=0x03; // Arm both OC1, IC0
        PORTT=0x00; // end of 5 us start pulse
    }
}
```

(10) Part f) The handler is executed about once a second, when the first echo is received.

```
#pragma interrupt_handler TC0handler()
void TC0handler(void){
    PutFifo(TCNT-First); // ranges from 333 33325 counts
    TMSK1=0x01; // disarm IC0
}
```