Solution 1.

Part a) The range of **u(n)** (500ns each) is 0 to 40000. As explained in Chapter 6, for simplicity reasons we will limit the range from about 100 to about 39900. This will allow enough time to service the output compare interrupt. The relationship is

u(n) = 400 • **U(t)**

Part b) First we convert the continuous integral into a discrete time

$$\mathbf{U(t) = U(t-\Delta t) + (R^*-R(t))\bullet\Delta t}$$

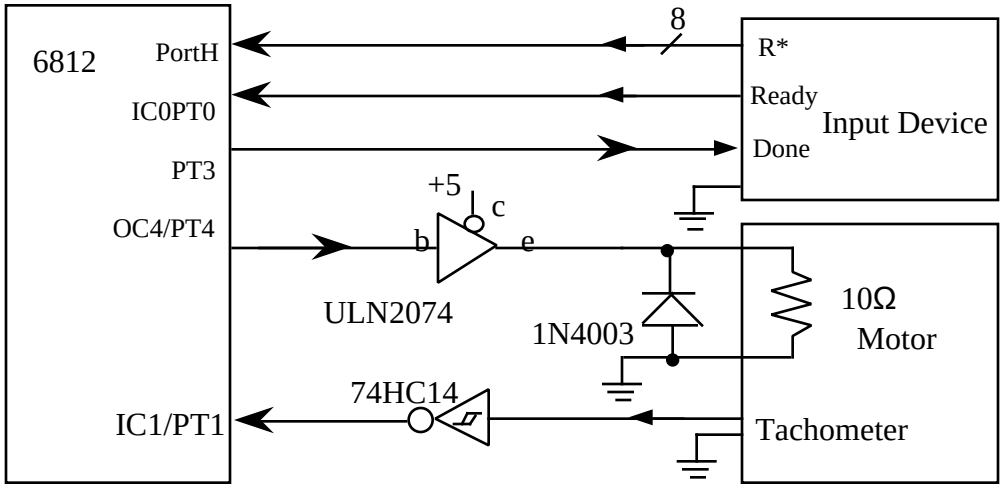
Next we use the relation found in part a)

$$\mathbf{u(n) = u(n-1) + 400\bullet(R^*-R(n))\bullet\Delta t}$$

At 50 Hz, $\Delta t = 0.02$ seconds,

$$\mathbf{u(n) = u(n-1) + 8\bullet(R^*-R(n))}$$

Part c) The ULN2074 has sufficient output current to drive the motor. The motor current will be 5V/10Ω=0.5A. The 74HC14 Schmitt-trigger will "clean up" the ugly waveform from the tachometer. The output on OC4/PT4 is used to generate the pulse-width modulated actuator control. The input on IC1/PT1 will use input capture to measure frequency. The desired frequency resolution is 100 Hz, which converts to a motor speed resolution of 1 rps. The Input Device is connected to IC0/PT0 so that the rising edge of Ready can generate an IC0 interrupt.



Part d) Ritual software including data structures. The main program executes the ritual, then performs other unrelated tasks. I.e., all processing occurs under interrupt control.

```
unsigned short count;  // raw count during frequency measurement
unsigned short Rstar;  // desired motor speed in rps
unsigned short R;      // current motor speed in rps
unsigned short u;      // actuator command 100 to 39900 counts
unsigned short low;    // u+low=40000
#define IC0F 0x01
#define IC1F 0x02
#define OC4F 0x10
#define Rate 40000  //  20 ms
void ritual(void) {
asm(" sei");          // make atomic
    DDRH=0x00;        // Port H is desired speed
    TIOS|=OC4F;       // enable OC4
    DDRT =0x18;       // PT4, PT3 are outputs
    TSCR=0x80;        // enable
    TMSK2=0x32;       // 500 ns clock
    TMSK1|=IC0F+IC1F+OC4F; // Arm IC0, IC1, OC4
    TC4=TCNT+Rate;    // First in 20 ms
    TCTL1|= 0x03;     // OC4/PT4 set on next interrupt
    TCTL4 = (TCTL4&0xF0)|0x05;
```

```
/* IC0F, IC1F both set on rising edges */
    count = 0;        // Set up for first
    u=100;            // initially off
    Rstar=0;
/* Set on the subsequent measurements */
    TFLG1=IC0F+IC1F+OC4F;  // clear IC0F, IC1F, OC4F
asm(" cli"); }
```

Part e) Show the interrupt software.

```
#pragma interrupt_handler TC2handler()
void TC2handler(void){
  count++;      // number of rising edges
  TFLG1=IC2F;}  // ack, clear IC2F
#pragma interrupt_handler TC3handler()
void TC3handler(void){
  PORTT |= 0x08;   // Done(PT3)=1
  Rstar=PORTH;
  PORTT ~= 0xF7;   // Done(PT3)=0
  TFLG1=IC3F;}  // ack, clear IC3F
#pragma interrupt_handler TC4handler()
void TC4handler(void){ long newu;
   if(TCTL1&0x04){      // PT4 is now high
      R = count>>1;     // R(n)=count/2  100 Hz units
      newu = u + 4*(2*Rstar - count);
      if(newu<100) u=100;
      else if(neww>39900) u=39990
      else u=newu
      low=40000-u;
      count = 0;       // Setup for next
      TC4=TC4+u;       // PT4 is 1 for u cycles
      TCTL1&=0xFE;}    // clear on next
   else {              // PT4 is now low
      TC4=TC4+low;     // PT4 is 0 for low cycles
      TCTL1|=0x03;}    // set on next interrupt
    TFLG1= OC4F;}      // Acknowledge
```

Solution 2. Interface an AD557 8-bit D/A directly to the MC68HC812A4 bus.

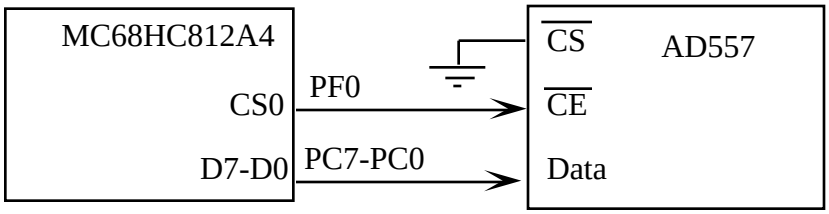
Part a) WDA = Write Data Available = (2 + 13, 1 + 14) = (106, 1 + 20)

Part b) WDR = (↑ $\overline{CE}$ -300ns, ↑ $\overline{CE}$ +10ns)

Part c) Circuit

$\overline{CS}$ =0

$\overline{CE}$ =CS0



Part d) WDA must overlap WDR 106 ≤ ↑ $\overline{CE}$ -300ns

Since the rise of CS0 occurs 10 ns after the fall of E 106 ≤ 1 + 10 -300ns or 396ns ≤ 1
Therefore there must be 3 cycle stretches so that the memory access time, 1, is 500 ns.

Solution 3. Design an analog to digital converter.