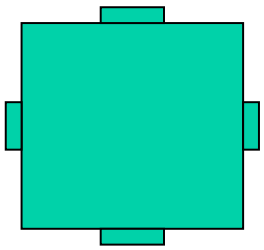
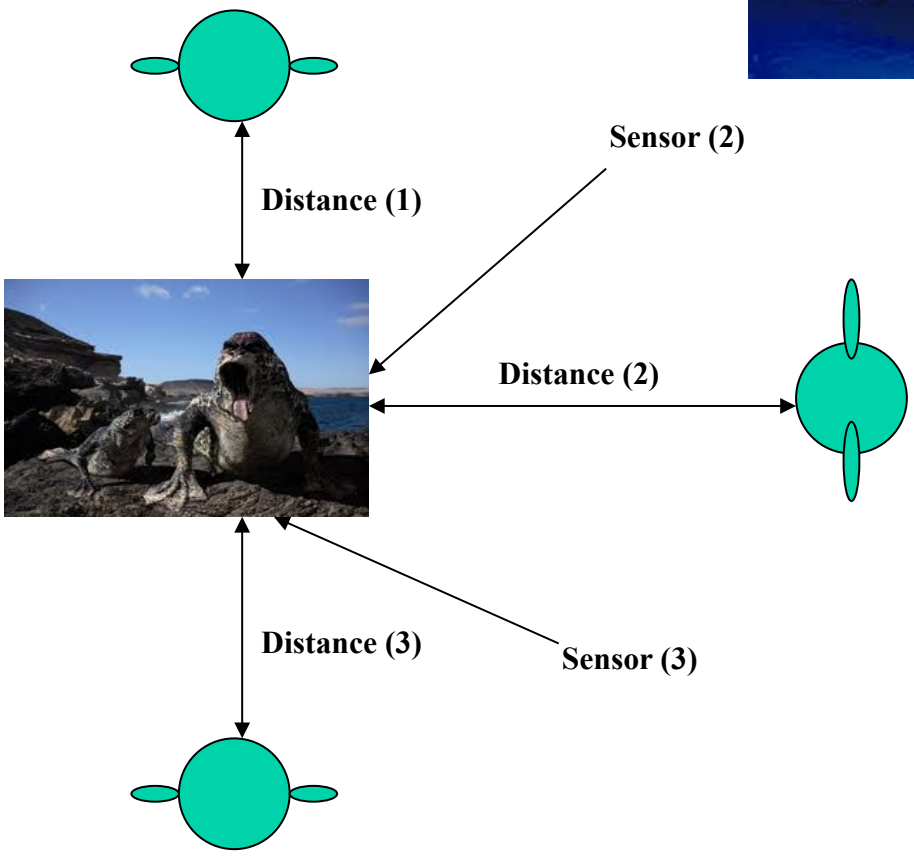
Vivisection - the Creature



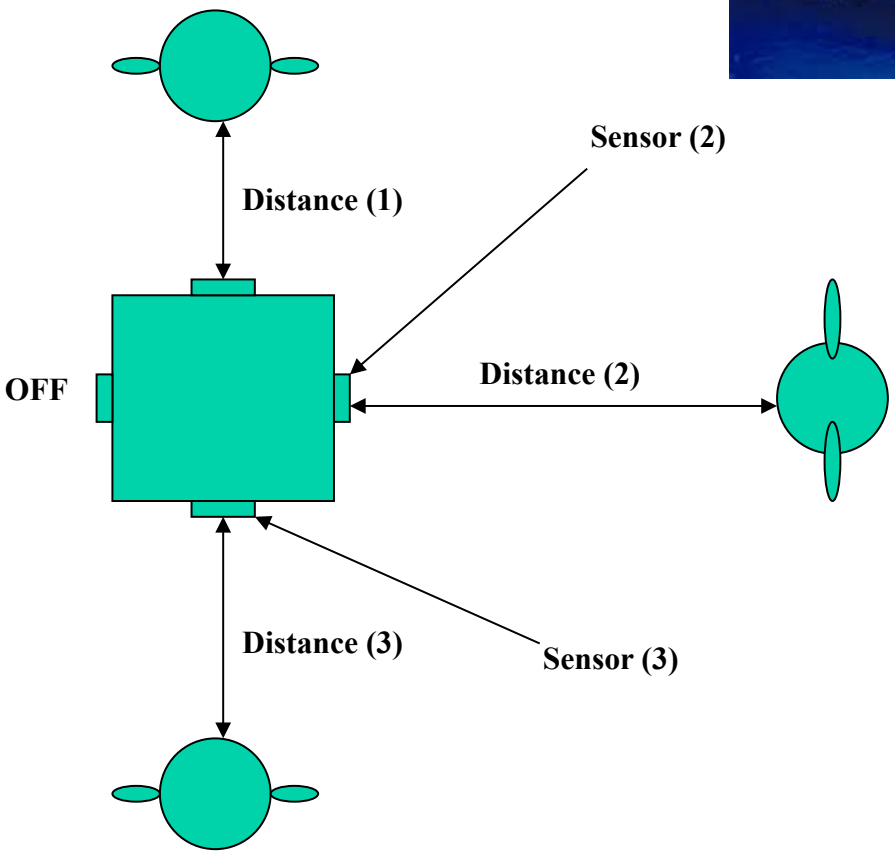
Vivisection Sensors



- **Sensors measure distance in centimeters.**
- **The embedded software uses the shortest distance to compute the behavior of the 8 tentacles.**

- **Sensor next to a wall would be turned OFF**

Vivisection Sensors

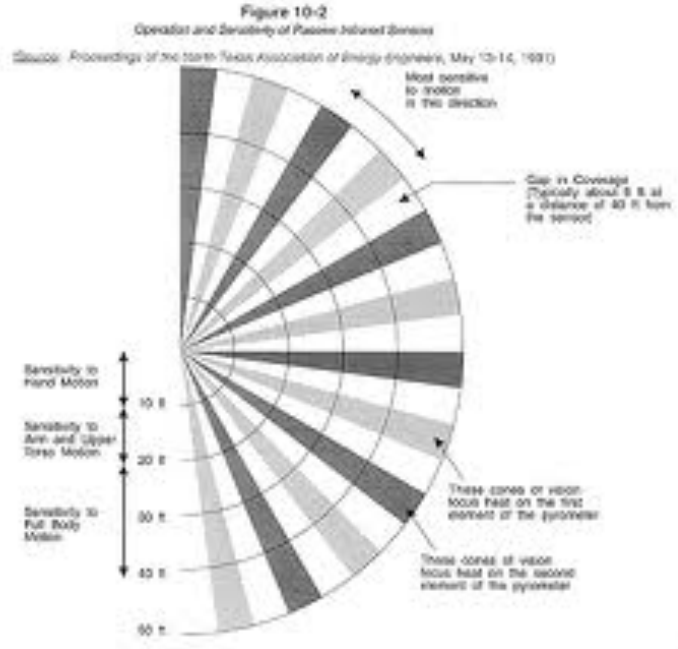


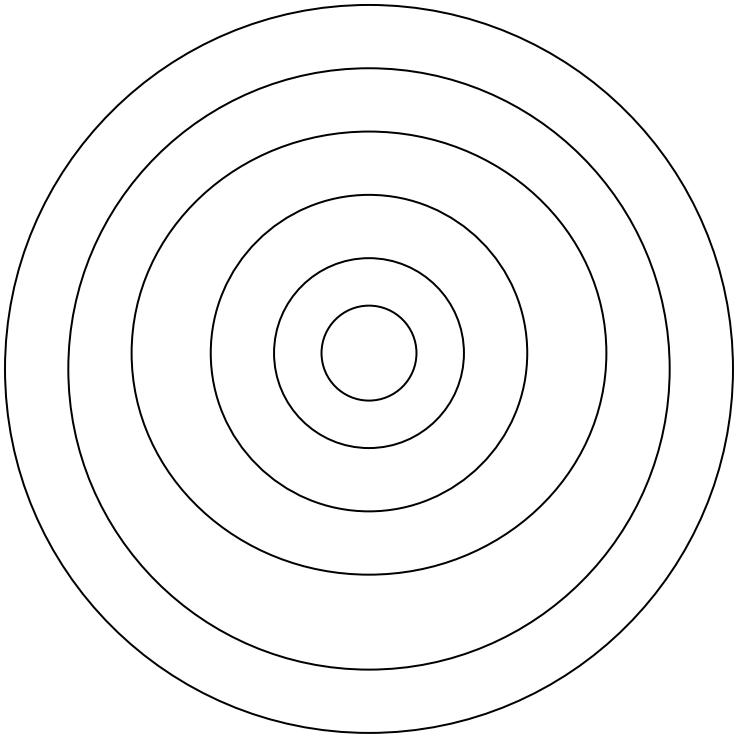
- **Sensors measure distance in centimeters.**
- **The embedded software uses the shortest distance to compute the behavior of the 8 tentacles.**

- **Sensor next to a wall would be turned OFF**

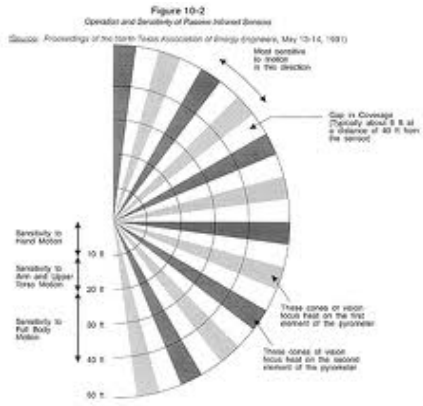
Vivisection Range Gates

Range Gates

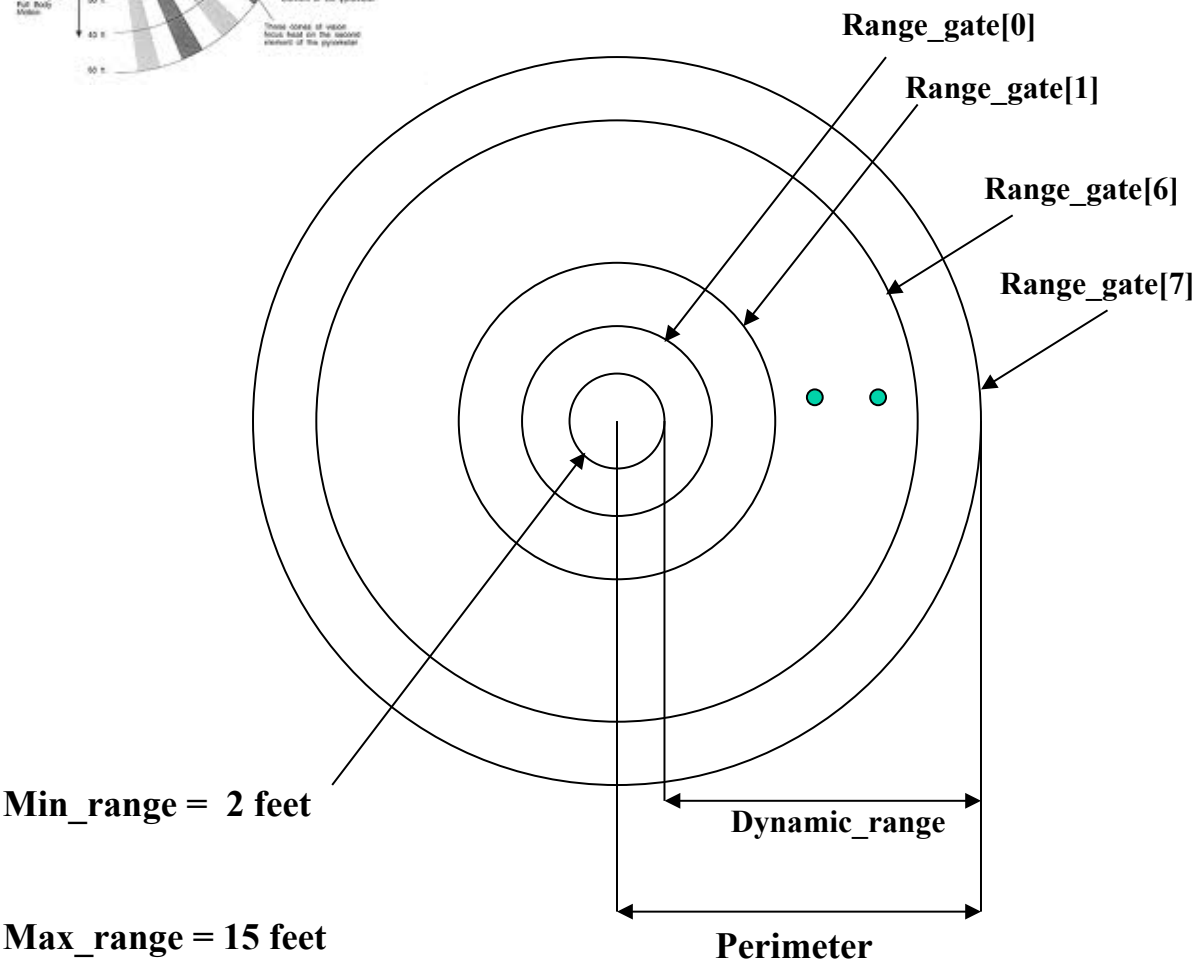




Vivisection Range Gates



Range Gates



Range_Gate Formula

Perimeter = max_range * (range_dial / 1024)

Dynamic_range = Perimeter - min_range

Range_gate[0] = min_range + dynamic_range / 8

Vivisection Software

Reading the Sensors

SensorTask( ) reads the AnalogIn Ch. 0 every 100msec, and loads the DAC value into the local variable = input. The for-loop updates the global variable = range when the input is less than the existing range.

Input: The analog input 0 is the voltage from a sonar device, which is perfectly linear with the distance from the target. Voltage range: 0 - 3.3V

Sample the target distance every 100msecs use the minimum distance value to set the digital_output_flags.

```
void SensorTask( void* p )
{
  (void) p;                        // unused variable
  int analog_input[4];
  int min,  x,  timeout;           //min = minimun value for all sensors


 range = perimeter;                // init range = perimeter
 InitializeRandomDigits();         //init RandomDigit[0] = 0, thru RD[7] = 7
 GenerateRandomDigits();           // initialize the 8 digit Random Number
 while( true )
 {
    ReadSustainDial();             // read sustain dial and calculate max_timeout
    All_Sensors_Off = 0;           //init all sensors to not be off

    analog_input[0] = AnalogIn_GetValue(0);  //read analog input Ch0, value is in cm
    analog_input[1] = AnalogIn_GetValue(1); //read analog input Ch1, value is in cm
    analog_input[2] = AnalogIn_GetValue(2); //read analog input Ch2, value is in cm
    analog_input[3] = AnalogIn_GetValue(3);  //read analog input Ch3, value is in cm

    if( (analog_input[0] > 750)&&(analog_input[1] > 750)&&
        (analog_input[2] > 750)&&(analog_input[3] > 750) )
    {
          All_Sensors_Off = 1;                 //all sensors off informs codependent
                                               //to cease activity
     }
```

Vivisection Software

Reading the Sensors

```
 min = perimeter +100;     //init minimum to be greater than perimeter
 for(x=0;x <= 3;x++)       //find minimum for the 4 sensors
{
       if(analog_input[x] < min)
       min = analog_input[x];
 }


if(min < range - 5)          //if target enters perimeter and keeps
{                            //moving forward at least 5cm before timeout
   range = min;              //set range = min reading
   timeout = 0;              //zero the timeout counter
}
else
   timeout+=20;              //if target stopped moving incr timeout

if(min >= perimeter)         //if target disappears
    timeout+=20;             //increment timeout because no target


if(timeout > max_timeout)    //if we reach timeout with no target
{
     timeout = 0;                    //clear timeout
     range = perimeter;              // set range to perimeter in order to reset outputs
     seed++;                         // increment global seed for random# generator
     GenerateRandomDigits();         //when target disappears generate new random#
 }

 Sleep( 20 );                  // wait 20msec so other tasks can run

}   // end while loop
}     // end SensorTask function
```

Vivisection Tentacles

Setting the Tentacle Flags

Task: TentacleOutputTask(void* p)

Description: TentacleOutputTask( ) calls 3 functions in order to set each of the tentacle_output [8] flags to ON or OFF

1. LoadRangeGateArray() takes the DAC value from AnalogIn(5) and derives the values for the range_gate[8]. The range_gate[8] set up the circles inside the perimeter, like the rings on a dart board. AnalogIn(5) is fed the voltage from the RangeGate Dial.

2. Range_Set_DigitalOutput_Flags() takes the range value from the sensors, from the SensorTask, and compares it to the range_gate[8] ing a switch/case statement. The digital_output[8] flags, which eventually turn ON or OFF each of the 8 tentacles, are set inside the switch statement.

3. TurnOn_DigitalOutputs()uses the digital_output[8] flags and the MAKE Firmware API function DigitalOut_SetValue(channel,value) to set each of the 8 digital output channels ON or OFF.

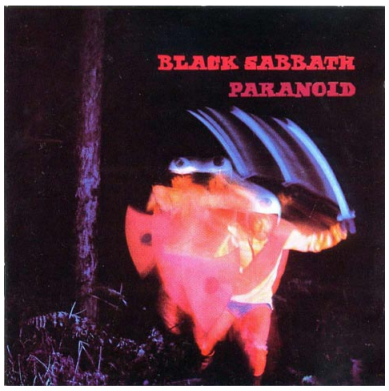
Input: The analog input 0 is the voltage from a sonar device, which is perfectly linear with the distance from the target. Voltage range: 0 - 3.3V

SensorTask samples the target range/distance every 100msecs the range is used to set the digital output flags the digital output channels are turned ON if there flag = 1

Vivisection Tentacles

Setting the Tentacle Output Flags

```
void TentacleOuputTask( void* p )
{
    (void) p;                                   // unused variable
    int inx,delay;
     int digital_out[8];

    InitializeRangeGates( );                    //initialize the range_gate[8]

    for(inx=0;inx<=7;inx++)
        DigitalOut_SetActive(inx,1);            //Set Activate all digital_out[8]

   while( true )
   {
        LoadRangeGateArray_UseFormula();
        GetMode();                              //get mode; 0 = paranoid, 1 = codependent
        if(mode==0)
            Range_Set_DigitalOutput_Paranoid(digital_out);          //Paranoid mode
        else
             Range_Set_DigitalOutput_Abandonment(digital_out);  //Abandonment mode

        TurnOn_DigitalOutputs(digital_out);

        Sleep(100);                             //allow other tasks to run once outputs are ON
  }

}    // end TentacleOutputTask
```

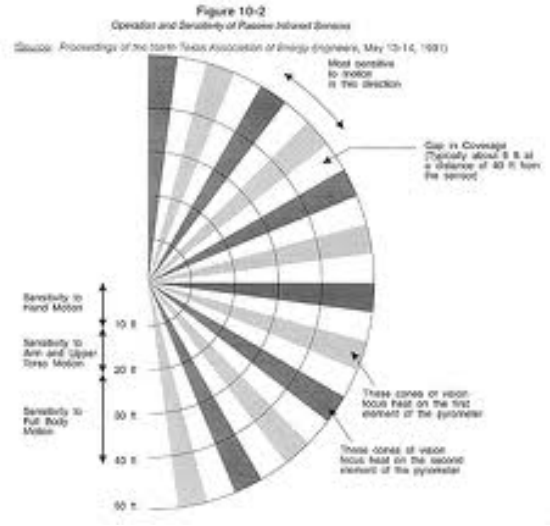


Vivisection Mode

Paranoid or Codependent?

- **Set Mode to Paranoid or Abandonment after reading the Mode Dial**

```
void GetMode(void)
{
//mode = DipSwitch_GetValueChannel(7); //read DipSw[7]on v1 board for mode

      if(AnalogIn_GetValue(6) < 500)  // v2 read AnalogIn 6
            mode = 0;
       else
            mode = 1;

}
```



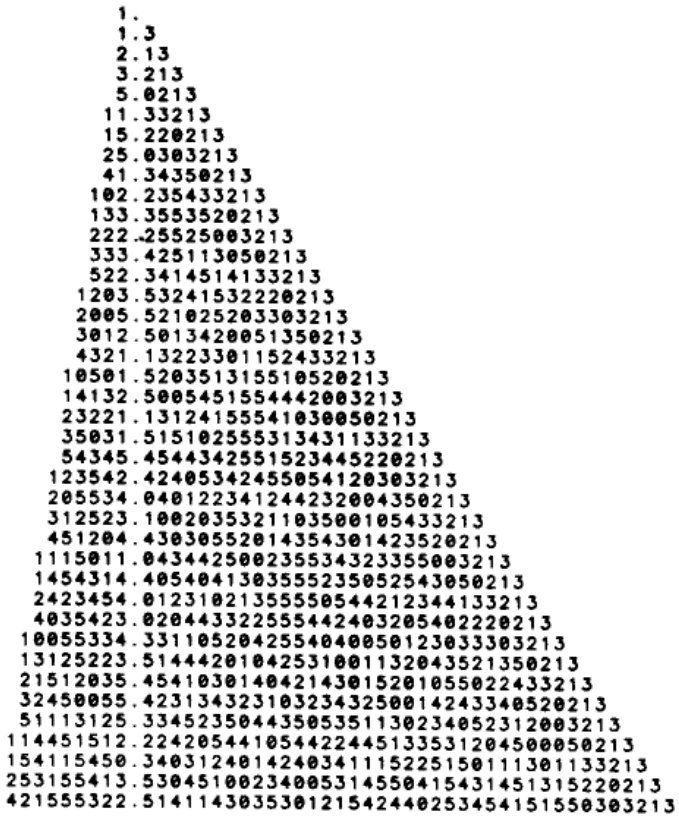
Vivisection Software

Calculate Range Gates

- **Calculate the range distances and load them into the Range Gate Array**

```
void LoadRangeGateArray_UseFormula(void)
{
    int range_dial = 500;                    //init to 500 as baseline
    float dynamic_range;
    int x;

    range_dial = AnalogIn_GetValue(5);       //read analog input Ch5, range dial:0 - 1023

    perimeter = (int)((float)max_range*((float)range_dial/1024.0)); // 450cm/500 = 0.9
   dynamic_range = (float)(perimeter - min_range);

    for(x=0; x<=7; x++)
    {
          range_gate[x] = min_range + (int)((x+1)*(dynamic_range/8.0));
    }

    perimeter = range_gate[7];               //re-load perimeter value in
                                             //case of any truncation error

}
```

Vivisection Software



Random Numbers

- **Generate 8 random digits (modulus 8) for the 8 tentacles**

```
void GenerateRandomDigits(void)
{
    int inx = 0;

    srand(seed);                          //use global_var timer_seed for seed
    for(inx = 0;inx<8;inx++)              // generate 8 random numbers
        RandomDigit[inx] = rand( )%8;
}
```

Vivisection Software

Turn on the Tentacles

• Turn-On the number of outputs that coresond to the target's range, and select each individual tentacle using a random number (modulus 8)

```
void Range_Set_DigitalOutput_Paranoid(int digital_out[8])
{
    int x;

    for(x=0; x<=7; x++)                    //clear all the digital output flags
        digital_out[x] = 0;

    if(range< range_gate[0])               //target within inner  range_gate[0]
    {
        digital_out[RandomDigit[7]] = 1;
        digital_out[RandomDigit[6]] = 1;
        digital_out[RandomDigit[5]] = 1;
        digital_out[RandomDigit[4]] = 1;
        digital_out[RandomDigit[3]] = 1;
        digital_out[RandomDigit[2]] = 1;
        digital_out[RandomDigit[1]] = 1;
        digital_out[RandomDigit[0]] = 1;
    }
    else if((range>= range_gate[0])&&(range< range_gate[1]))  //
    {
         digital_out[RandomDigit[7]] = 1;
         digital_out[RandomDigit[6]] = 1;
         digital_out[RandomDigit[5]] = 1;
         digital_out[RandomDigit[4]] = 1;
         digital_out[RandomDigit[3]] = 1;
         digital_out[RandomDigit[2]] = 1;
         digital_out[RandomDigit[1]] = 1;
         digital_out[RandomDigit[0]] = 1;
     }
    else if((range>=range_gate[1])&&(range<range_gate[2]))
    {
        digital_out[RandomDigit[5]] = 1;
        digital_out[RandomDigit[4]] = 1;
        digital_out[RandomDigit[3]] = 1;
        digital_out[RandomDigit[2]] = 1;
        digital_out[RandomDigit[1]] = 1;
        digital_out[RandomDigit[0]] = 1;
     }
```

Vivisection Software

Turn on the Tentacles

• Turn-On the number of outputs that corresond to the target's range, and select each individual tentacle using a random number (modulus 8)

```
void Range_Set_DigitalOutput_Paranoid(int digital_out[8])
{
      < continued >


       else if((range >=  range_gate[2] ) && ( range < range_gate[3]))
       {
              digital_out[RandomDigit[5]] = 1;
              digital_out[RandomDigit[4]] = 1;
              digital_out[RandomDigit[3]] = 1;
              digital_out[RandomDigit[2]] = 1;
              digital_out[RandomDigit[1]] = 1;
              digital_out[RandomDigit[0]] = 1;
        }
```

--- continue to load digital_out ----