

Vacuum Extraction Tube System (VETS)

The VETS is a system of large diameter vacuum tubing that runs from 2 remote stations to the PACS installed in the garage. The PACS (Pneumatic Actuated Can Smasher), described in a previous Gadget Freak article ([Case#85, 5/15/06 "Ed had a Fondness for Orange Crush"](#)), was modified to accommodate the delivery system by replacing the magazine with a new section, referred to here and in the Control Program as the Stack. Some rewiring was also required since a single microcontroller now operates both the PACS and the VETS.

One remote station is on the back patio and the other is on the main support column in the garage. The tubing connects at a 'Y' fitting right above the PACS. The entry stations are identical. Each consists of a steel compartment with a door for depositing the can to be recycled. This compartment is part of a cylindrical section of steel tubing that is connected to the main vacuum transport tubing. The remote stations also have a status display panel that contains 3 indicator lights and a mode switch. In addition, there is a flow control door at the very end of each remote station that is used to control the flow of air (and hence the can) through the tube. The door is controlled by a pneumatic actuator, connected via an air line to a solenoid valve controlled by the PIC.

The source of vacuum is a centrifugal blower mounted in the compressor shed and connected by a 2" diameter tube near the exit above the PACS. The blower has a pneumatic actuator that controls a butterfly valve on the vacuum side to enable applying or removing vacuum rapidly to the system. The actuator is controlled by the PIC through a pneumatic solenoid valve. The power for the blower is controlled through an SSR mounted in the motor housing. The can exits through a vertical section of tubing that is lined up with the entrance to the PACS. Both remote entry stations and the exit have pneumatic actuators that control air flow doors on each. The doors allow the computer to control the flow of air through the individual tubes permitting the extraction of a can from either station automatically.

The PACS consists of a precision bore crushing cylinder and an aluminum piston that is connected to a 4" diameter pneumatic ram. The crushing cylinder is fed through a transition tube. The cylinder, piston and ram are mounted horizontally to a rigid steel frame. The transition tube is a vertical section of tubing mounted directly under the exit of the VETS. This "transitions" to the horizontal exit above the crushing cylinder through a large radius elbow. The vertical section of the transition tube has 2 pneumatic actuators mounted directly above each other at a distance of 1 can length apart, and a 3rd actuator which controls the exit door. Their purpose is to sequence the loading of cans into the breech. There are 2 optical proximity sensors. One looks through the transition tube to detect cans waiting to be loaded and the other looks through the breech to sense a can in the smashing position. Some time, the 1st of these sensors will be replaced with two sensors, one in each tube upstream of the Y connection to better accommodate actuator delays and the high arrival speed of the cans.

There are a few items on the schematics whose functions are not yet implemented. One is the warning light. Others are the three maintenance switches connected to PORTE.0,1,2 of the PIC. Another is Multi-Can Mode, which is only anticipated by a position of the Mode switches.

Microcontroller Program:

The source file is "Vacuum Extraction Recycling Facility Ver 1[1].01.pbp.txt". PBP stands for "PIC BASIC PRO" BY MICROENGINEERING LABS. This file is viewable and editable with any text editor. Remove the ".txt" extension from the filename before compilation.

List of Peripherals:

Optical Sensors (in downstream sequence):

Can_Sta_1 (Can sensor for Station 1)
Can_Sta_2 (Can sensor for Station 2)
Arrival_Sensor
Stack_Sensor
Breech_Sensor

Total: 5 (so far, see text)

Microswitches:

Door_Open_sw (one for each station)

Total: 2

Actuators:

Act_Sta_1 (Flow Control door for Station 1)
Act_Sta_2 (Flow Control door for Station 2)
Vac_Valve (butterfly valve)
Stack_Hold_Actuator
Ready_Pos_Actuator
Exit_Door
Crushing_Ram (valve for this one has 2 coils)

Total: 7

Solenoid air valves:

The above 7, plus:

Main_Air_Valve

Total: 8

```

*****
* Name   : Can Crusher Control Program          *
* Author : Edward A. Nauman                   *
* Notice : Copyright (c) 2006                 *
*       : All Rights Reserved                 *
* Date   : 1/25/2006                          *
* Version : 4.0                               *
* Notes  :                                     *
*       :                                     *
*****

```

```

*****
***** General Description *****
*****

```

```

' This program is the operational software for a fully automatic
' pneumatic can crusher with a vacuum extraction system.
' The system is fully automatic. All that is required of the user to operate
' the system is flip a switch, open the door, place the can in and close the door.
' When the cycle is complete (~1 sec) a ready light will illuminate and he or she
' may either place another can in the machine or turn it off.

```

```

*****
***** Hardware Description *****
*****

```

```

' The complete system consists of 2 subsystems that are controlled by a single
' PIC16F877 microcontroller and associated circuitry. The first subsystem is
' Vacuum Extraction Tube System (VETS). The other is the Pneumatic Actuated
' Can Smasher (PACS). The VETS provides transport for the cans from 2 locations
' to the PACS installed in the garage. The PACS smashes the cans and deposits them
' in a plastic trash bag for recycling. The two subsystems are described below

```

```

' ***** VETS Description *****

```

```

' The VETS is a system of large diameter vacuum tubing that runs from the PACS
' installed in the garage, to 2 remote stations. One remote station is on
' the back patio and the other is on the main support column in the garage.
' The tubing connects at a 'Y' fitting right above the PACS.
' The entry stations are identical. Each consists
' of a steel compartment with a door for depositing the can to be recycled. This
' compartment is part of a cylindrical section of steel tubing that is connected
' to the main vacuum transport tubing. The remote stations also have
' a status display panel that contains 3 indicator lights and a mode switch.
' In addition, there is a flow control door at the very end
' of each remote station that is used to control the flow

```

' of air (and hence the can) through the tube.
' The source of vacuum is a centrifugal blower mounted in the compressor shed
' and connected by a 2" diameter tube near the exit above the PACS.
' The blower has a pneumatic actuator that controls a butterfly valve
' on the vacuum side to enable applying or removing vacuum
' rapidly to the system.
' The actuator is controlled by the PIC through a pneumatic solenoid valve.
' The power for the blower is controlled through an SSR
' mounted in the motor housing. The can exits through a vertical section
' of tubing that is lined up with the entrance to the PACS. Both remote
' entry stations and the exit have pneumatic actuators that control air flow
' doors on each. The doors allow the computer to control the flow
' of air through the individual tubes permitting the extraction of a can
' from either station automatically.

***** PACS *****

' The PACS consists of a precision bore crushing cylinder and an aluminum
' piston that is connected to a 4" diameter pneumatic ram. The crushing
' cylinder is fed through a transition tube. The cylinder, piston and ram
' are mounted horizontally to a rigid steel frame. The transition tube
' is a vertical section of tubing mounted directly under the exit of the VETS.
' This "transitions" to the horizontal exit above the
' crushing cylinder through a large radius elbow. The vertical section
' of the transition tube has 2 pneumatic actuators mounted directly
' above each other at a distance of 1 can length apart.
' Their purpose is to sequence the loading of cans into the breech. There are
' 2 optical proximity sensors. One looks through the transition tube
' to detect cans waiting to be loaded and the other looks through the
' breech to sense a can in the smashing position.

' *****Operational Description*****
' *****

' When the user flips the mode switch into the single can mode, the PIC
' closes all of the air flow control doors, the vacuum butterfly valve,
' and starts the vacuum pump (blower). The ready light is turned on.
' When the user opens the door, the door open indicator is turned on
' and the ready light is turned off.
' The MCU waits until it senses a can has been placed in the tube.
' Once a can has been placed in the tube and the door is closed, the MCU
' turns off the door open light, turns on the wait light and
' opens the butterfly valve to depressurize the tubes. Then the MCU opens
' the flow control door on the remote station containing the can.
' This allows the air (and hence the can) to flow from the remote station
' toward the PACS. As the can passes an optical sensor about 5 feet from
' end, the MCU closes the butterfly valve, opens the exit door, but
' leaves the vacuum pump on in case the user has more cans.

' The can coasts the last few feet under it's momentum and drops
 ' into the PACS. The operation of the PACS is explained further
 ' down in this document. Once the PACS detects the
 ' can in its magazine, the MCU closes the flow control
 ' doors on the remote station and the exit to prepare for the next can.
 ' The wait light is left on until the PACS has finished crushing the can.
 ' After the PACS is through, the wait light is turned off and the ready
 ' light is turned on and the system is ready to accept another can.
 ' At this point the MCU starts a timer. If no action is detected
 ' within 1 minute the MCU turns off the vacuum and waits for another can.
 ' The user should put the mode switch in the off position when through.
 ' If the user forgets to set the mode switch to off, the MCU will close
 ' main air supply valve after 5 minutes and go into standby mode.
 ' If one of the remote doors is opened while in standby mode, the
 ' MCU turns the door open light on, opens the main air supply valve and
 ' turns on the vacuum pump. The system is now ready to accept a can.

*****Multi-Can Mode*****

' When the user places the mode switch in the Multi-Can Mode, the
 ' operation is the same as above for the first can. However, when the
 ' first can drops out the exit into the PACS, the MCU closes the
 ' exit door but leaves the flow control door on the remote station open.
 ' This is to allow the user to feed cans in from the bottom without
 ' having to open the door. The wait light remains on and the butterfly
 ' valve remains closed until the PACS has finished crushing the can it has.
 ' Once the PACS has finished, the MCU turns the ready light on,
 ' the wait light off and waits for the sensor to indicate another can is
 ' ready to go. When the user inserts the next can into the flow control
 ' opening at the bottom, the sensor notifies the MCU. The MCU opens the
 ' butterfly valve and the can is literally sucked out of the users hand.
 ' The process repeats as long as the user keeps feeding cans.
 ' After 2 minutes elapse with no activity, the MCU closes the flow control
 ' and exit doors, turns off the main air supply valve and vacuum pump.
 ' The system goes into standby mode at this point.

 ***** Main Program Operation *****

 *****Item definitions*****

setup:

```

DEFINE OSC 20
DEFINE HSER_RCSTA 90h
DEFINE HSER_TXSTA 24h
DEFINE HSER_BAUD 9600
DEFINE HSER_PACING 1000
DEFINE DEBUG_REG PORTC
DEFINE DEBUG_BIT 6
DEFINE DEBUG_BAUD 9600
DEFINE DEBUG_MODE 0
DEFINE SHIFT_PAUSEUS 100
define CHAR_PACING 300
define DEBUG_PACING 5000

```

```

TRISA = %11110001      'Sets the I/O pins for the MCU
TRISB = %11001001
TRISC = %10011000
TRISD = %01111000
ADCON1= %00000110     'Sets the ADC status (not used)

```

```

*****
***** Define some alias names *****
*****

```

```

Arrival_Sensor  VAR PORTA.0 ' Arrival sensor. 0 = Can detected
Vac_Pwr         VAR PORTA.1 ' Blower Power Signal. 1 = Blower on
Vac_Valve       VAR PORTA.2 ' Vacuum butterfly valve. 1 = Valve open (vacuum)
Exit_Door       VAR PORTA.3 ' Exit door actuator. 1 = door open
Stack_Sensor    VAR PORTA.4 ' Stack sensor. 0 = Can present
Breech_Sensor   VAR PORTA.5 ' Breech Sensor. 0 = Can in breech
Can_Sta_2       var PORTB.0 ' Station 2 can sensor. 0 = Can present
Act_Sta_2       var PORTB.1 ' Station 2 flow control door. 1 = door open
Crushing_Ram    var PORTB.2 ' Crush solenoid valve. 1 = crush
'              PORTB.3 is not used
Retract_Ram     var PORTB.4 ' Retract crushing ram. 1 = retract
Stack_Hold_Actuator var PORTB.5 ' Stack holding ram. 1 = hold stack
Ready_Pos_Actuator var PORTC.0 ' Ready position ram. 1 = release can
Main_Air_Valve  var PORTC.1 ' Main air supply valve. 1 = open
Warning_Light   var PORTC.2 ' Warnig Light
'              PORTC.3 is not used
'              PORTC.4 is not used
LCD_Data        var PORTC.5 ' Data Line to LCD
Tx_Data         var PORTC.6 ' RS232 transmit line (to PC)
Rx_Data         var PORTC.7 ' RS232 recieve line (from PC)
Door_Open_Ind   var PORTD.0 ' Door open indicator (output) 1 = door open
Wait_Ind        var PORTD.1 ' Wait indicator (output) 1 = wait

```

```

Ready_Ind      var PORTD.2 ' Ready indicator (output) 1 = ready
Door_Open_Sw   var PORTD.3 ' Door open switch (input) 0 = door open
Multi_Mode     var PORTD.4 ' Multiple can mode (input) 0 = multi mode
Single_Mode    var PORTD.5 ' Single can mode (input) 0 = single can mode
Can_Sta_1      var PORTD.6 ' Station 1 can sensor. 0 = Can present
Act_Sta_1      var PORTD.7 ' Station 1 flow control door. 1 = door open

```

```

*****

```

```

***** Define some Constants *****

```

```

*****

```

```

TURN_OFF con 0 'Constant for indicators and crushing ram solenoids

```

```

TURN_ON con 1 'Constant for indicators and crushing ram solenoids

```

```

YES con 0

```

```

NO con 1

```

```

LOAD con 1

```

```

HOLD con 0

```

```

RELEASE CON 1

```

```

RETRACT con 1

```

```

RELEASE_STACK con 0

```

```

HOLD_STACK con 1

```

```

SHUT CON 0

```

```

OPEN CON 1

```

```

CLOSED con 1

```

```

NOT_CLOSED con 0

```

```

CRUSH con 1

```

```

*****

```

```

***** Define some Variables *****

```

```

*****

```

```

Arrival_Sensor_Error var bit 'used to show that the Arrival_Sensor is on

```

```

Stack_Sensor_Error var bit 'used to show that the Stack_Sensor is on

```

```

Breech_Sensor_Error var bit 'used to show that the Breech_Sensor is on

```

```

Ctr var word

```

```

Mode var byte

```

```

i var byte

```

```

Status_Word Var byte 'Variable to hold the switch & sensor status

```

```

Station var byte 'Variable to hold the number of the active remote station

```

```

Sensor_Status var byte 'Variable used to check the status of the PAC sensors.

```

```

***** Bit Designations for Status Word *****

```

```

***** All bits are active low *****

```

```

'Status_Word.0 = Single_Mode

```

```

'Status_Word.1 = Multi_Mode

```

```

'Status_Word.2 = Can_Sta_1

```

```

'Status_Word.3 = Can_Sta_2

```

```

'Status_Word.4 = Door_Open_Sw

```

```
'Status_Word.5 = Stack_Sensor
'Status_Word.6 = Breech_Sensor
'Status_Word.7 = Arrival_Sensor
```

```
*****
*****Initialize all rams to starting positions*****
*****
```

Initialize:

```
'debug $FE,1
debug "Initalizing",10,13
main_air_valve = Turn_on 'Turn on the air to retract the rams
pause 2000
Crushing_Ram = Turn_Off 'Set the crush solenoid to off (no current)
pause 100 'solenoid response time
Retract_Ram = RETRACT 'command the valve to retract the ram
pause 500 'Actuator retract time
Retract_Ram = Turn_Off 'Turn current to solenoid off
pause 100 'solenoid response time
Ready_Pos_Actuator = hold 'Block the ready position
pause 100 'solenoid response time
Stack_Hold_Actuator = release_stack 'Open the chute for loading
pause 100
Act_Sta_1 = SHUT
Act_Sta_2 = SHUT
Exit_Door = SHUT
Vac_Valve = shut
vac_pwr = turn_off
gosub lamp_test
main_air_valve = Turn_off
*****
***** Main Program *****
*****
```

Main:

```
debug "Main Program - beginning",10,13
gosub Check_Mode 'Check the mode switches for off or error condition.
```

```
iF Multi_Mode = 0 and Single_Mode = 1 then
  Main_Air_Valve = Turn_On
  goto Multi
endif
```

```
iF Multi_Mode = 1 and Single_Mode = 0 then
  Main_Air_Valve = Turn_On
  goto Single
else
```

```

goto Main
endif

```

Single:

```

Act_Sta_1 = Shut
Act_Sta_2 = Shut
Exit_Door = shut
Vac_Valve = shut 'Close the butterfly valve
Vac_Pwr = TURN_ON 'Turn on the vacuum pump

```

```

if door_Open_Sw = not_closed then
  gosub turn_on_door_open_ind
  gosub check_mode 'Check the mode switches for off or error condition.
endif

```

'Check for the proper run conditions. If they exist, branch to the run subroutine.
 'If not exist, check the mode switches and PAC sensors. If mode and sensors are
 'correct, loop back to main until the run conditions exist or mode switches
 'are off or in error.

```

gosub Get_Status
debug "Status_Word = ",dec status_word,10,13
pause 1000
select case Status_Word
  case %1111010 'Door is closed, Can in remote #1, single mode,
    Station = 1 'all other sensors correct. 'Set remote variable to 1.
    Gosub Run
    gosub Turn_On_Wait_Ind
  case %11110110 'Door is closed, Can in remote #2, single mode,
    Station = 2 'all other sensors correct. 'Set remote variable to 2.
    Gosub Run
    gosub Turn_On_Wait_Ind
  case %11111110
    gosub Turn_On_Ready_Ind
  case else
    gosub Sensor_Check 'Check the breech, stack and arrival sensors.
    gosub check_mode 'Check the mode switches for off or error condition.
end select
goto main

```

Multi:

```

debug "Multi_Mode Selected",10,13
goto main

```


'It also monitors the mode switches so that the program can be reset in the
'event the sensor is not triggered.

Wait_For_Arrival_Loop:

```

gosub Check_Mode
while Arrival_Sensor = no
  goto Wait_For_Arrival_Loop:
wend
exit_door = open
vac_valve = shut
pause 100
Act_Sta_1 = Shut
Act_Sta_2 = Shut
return

```

```

*****
***** Crush It *****
*****

```

'This subroutine operates the VETS once the can has triggered the
'arrival sensor. The sub routine waits for the stack sensor to be triggered
'while checking the mode switch for an error or off condition. Once the stack
'sensor is triggered, it crushes the can and checks the stack sensor for a
'second can. If there is no can, it returns to where it was called from.
'If there is one, it crushes it and returns to where it was called from.

Crush_It:

```

gosub Check_Mode
if Stack_Sensor = no then crush_it 'if stack sensor has not been triggered keep checking.
exit_door = shut
stack_hold_actuator = hold_stack 'hold the stack in case there are 2 cans
pause 100
ready_pos_actuator = load 'Load the can.
pause 100
Wait_For_Breech_Sensor:
if Breech_sensor = yes then
  PAUSE 100
  crushing_ram = crush 'Time for can to settle
  pause 100
  crushing_ram = turn_off 'turn off current to crushing ram solenoid
  pause 500 'time for ram extension
  retract_ram = retract
  pause 50
  retract_ram = turn_off 'turn off current to crushing ram retract solenoid
  ready_pos_actuator = hold
  pause 50

```

```
stack_hold_actuator = release_stack
pause 200
```

```
if stack_sensor = yes then 'If there is another can then crush it.
  stack_hold_actuator = hold_stack 'hold the stack in case there are more cans
  pause 100
  ready_pos_actuator = load 'Load the can.
  PAUSE 100
  crushing_ram = crush 'Time for can to settle
  pause 100
  crushing_ram = turn_off 'turn off current to crushing ram solenoid
  pause 500 'time for ram extension
  retract_ram = retract
  pause 50
  retract_ram = turn_off 'turn off current to crushing ram retract solenoid
  ready_pos_actuator = hold
  pause 50
  stack_hold_actuator = release_stack
endif
```

```
else
  debug "Waiting for breech sensor",10,13
  goto wait_for_breech_sensor 'If the can has not triggered the breech
    'sensor yet , go check it again.
return
endif
return
```

```
*****
*****Check Mode*****
*****
```

'This subroutine checks to see if the error condition (both remote station
'mode switches selecting different modes at the same time)
'has been cleared. If not, it flashes all the indicator lights and loops until
'cleared. If modes switches are in off position, go to the beginning of the main.

Check_Mode:

```
'debug "Check power switches on both remote stations",10,13
if Multi_Mode = yes and Single_Mode = yes then 'if error is not cleared, loop until it is.
  main_air_valve = Turn_Off
  Vac_pwr = turn_off
  gosub Lamp_Test
  goto check_mode
```

```
endif
if multi_mode = no and single_mode = no then 'If mode switches are off,
  main_air_valve = Turn_Off          'turn shit off and start over.
  Vac_pwr = turn_off
  Act_Sta_1 = SHUT
  Act_Sta_2 = SHUT
  Exit_Door = SHUT
  Vac_Valve = shut
  gosub lamp_test
  goto Standby
else
  return
endif
```

```
*****
***** Standby*****
*****
```

'This subroutine checks the mode switches and waits for an on condition.

Standby:

```
if single_mode = yes or multi_mode = yes then main
goto standby
```

```
*****
*****Lamp Test Subroutine*****
*****
```

Lamp_Test:

```
Warning_Light = 1
Door_Open_Ind = 1
Wait_Ind = 1
Ready_Ind = 1
```

pause 1000

```
Warning_Light = 0
Door_Open_Ind = 0
Wait_Ind = 0
Ready_Ind = 0
```

pause 500

```
debug "End of lamp test",10,13
return
```

```

*****
*****Get Status Subroutine*****
*****

```

'This subroutine reads the switches and sensors into a status
'word and returns.

Get_Status:

```

Status_Word.0 = Single_Mode
Status_Word.1 = Multi_Mode
Status_Word.2 = Can_Sta_1
Status_Word.3 = Can_Sta_2
Status_Word.4 = Door_Open_Sw
Status_Word.5 = Stack_Sensor
Status_Word.6 = Breech_Sensor
Status_Word.7 = Arrival_Sensor
return

```

```

*****
*****Sensor_Check*****
*****

```

'This subroutine checks the breech, stack and arrival sensors. If
'any are active, the program returns an error message and loops until
'the error is cleared or the mode switches are off.

Sensor_Check:

```

debug "Sensor Check Loop",10,13
gosub get_status 'read the sensors and switches
debug "Status_Word = ",bin status_word,10,13
Sensor_Status = Status_Word & %11100000
if sensor_status < 224 Then
  Stack_Sensor_Error = Status_Word.5
  Breech_Sensor_Error = Status_Word.6
  Arrival_Sensor_Error = Status_Word.7
  if Stack_Sensor_Error = yes then
    debug "Check Stack Sensor ",10,13
  endif
  if breech_Sensor_error = yes then
    debug "Breech Sensor Error",10,13
  endif
  if arrival_Sensor_error = yes then

```

```

    debug "Arrival Sensor Error",10,13
endif
gosub check_mode
goto sensor_check
else
    return
endif

```

```

'*****
'***** Turn_On_Wait_Ind *****
'*****

```

"This subroutine Turns on the wait indicator and turns off the others.

```

Turn_On_Wait_Ind:
Ready_ind = turn_off
door_open_ind = turn_off
wait_ind = Turn_on
return

```

```

'*****
'***** Turn_On_Ready_Ind *****
'*****

```

"This subroutine Turns on the Ready indicator and turns off the others.

```

Turn_On_Ready_Ind:
Ready_ind = turn_on
door_open_ind = turn_off
wait_ind = Turn_off
return

```

```

'*****
'***** Turn_On_Door_Open_Ind *****
'*****

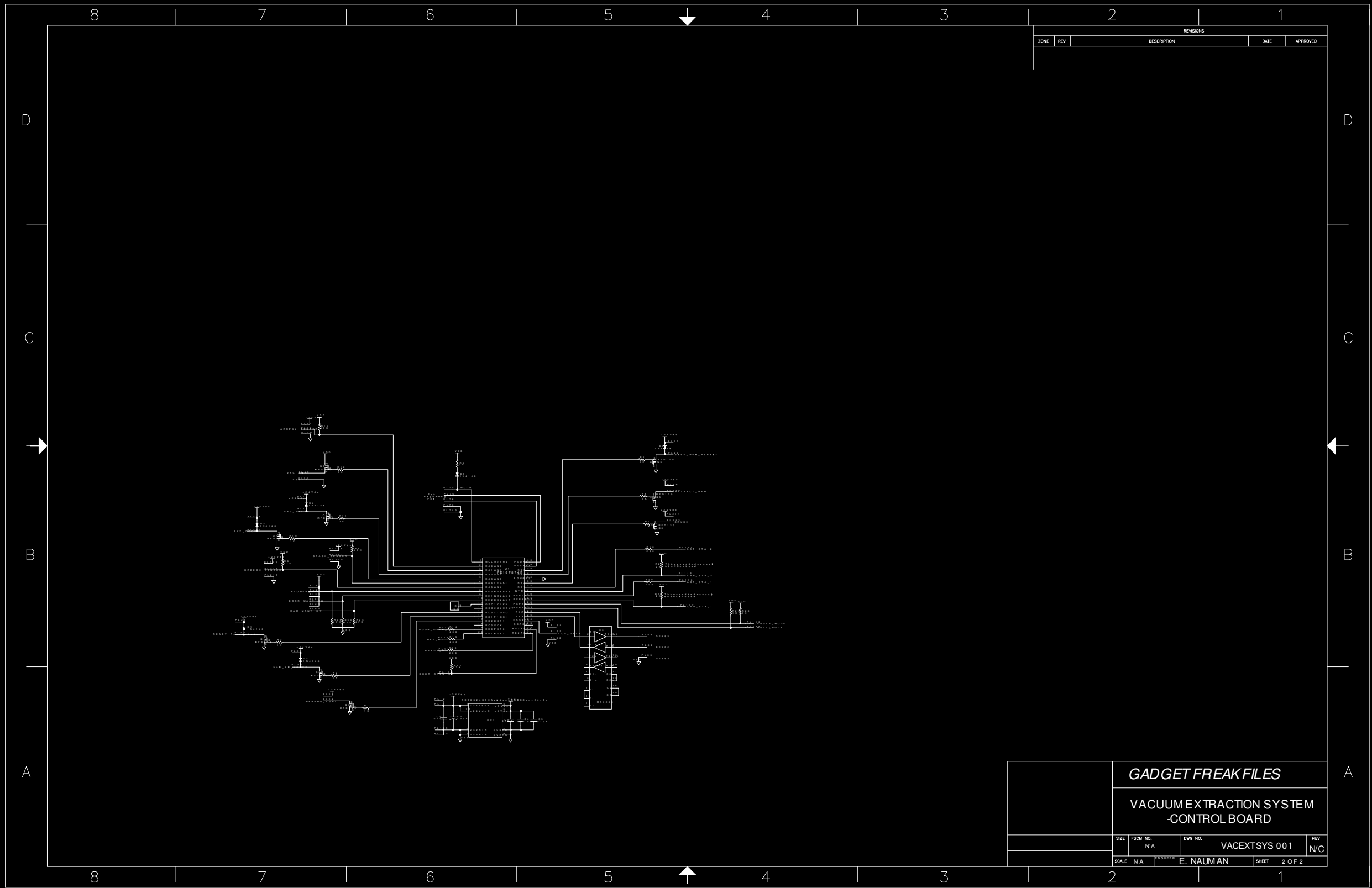
```

"This subroutine Turns on the wait indicator and turns off the others.

```

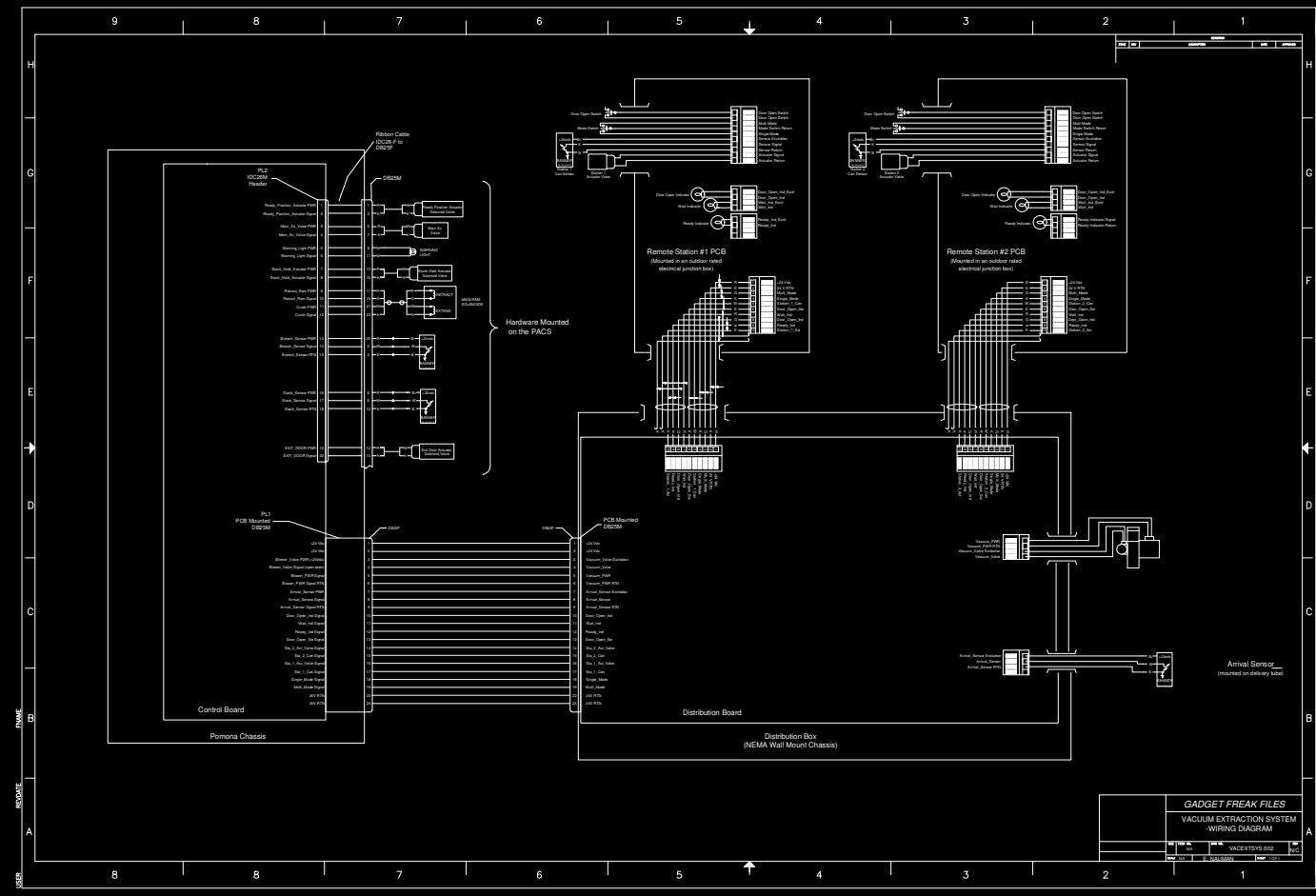
Turn_On_Door_Open_Ind:
Ready_ind = turn_off
door_open_ind = turn_on
wait_ind = Turn_off
return

```



REVISIONS				
ZONE	REV	DESCRIPTION	DATE	APPROVED

GADGET FREAK FILES			
VACUUM EXTRACTION SYSTEM CONTROL BOARD			
SIZE	FSCM NO.	DWG NO.	REV
		VACEXTSYS 001	N/C
SCALE		DESIGNED BY	SHEET
N/A		E. NAUMAN	2 OF 2



QTY	REF DES	ALLIED P/N
13	R1-R7,R10-R12,R22,R23	895-2722
7	R8,R9,R13,R17,R24-R26	895-2859
5	R14,R15,R16,R18,R20	895-2740
6	D1-D6	431-0614
1	U1	383-0489
1	U3	773-0043
9	Q1-Q9	503-0398
1	PL2	512-3804
1	PL1	720-6190
1	PL8	720-6120
3	PL4-PL6	UNKNOWN
1	U2	N/A
1	PL7	UNKNOWN

DESCRIPTION

1k Surf Mount Resistor, 0805 case
4.7k Surf Mount Resistor, 0805 case
200 Ohm Surf Mount Resistor, 0805 case
1N4148 DIODE
PIC16F877 MCU
MAX233 RS232 Interface Chip
N-Channel Hexfet
26 Pin Header Connector
25 Pin Male Subminiature D CONNECTOR
9 Pin Male Subminiature D CONNECTOR
3 Pin Molex PCB Header Connector
Cardinal Programmable Oscillator (CPP-C 7 T A5 BP TC)
10 Pin Header Connector