Figure1. The core of this circuit is straight off the propeller demo board schematic from Parallax. I have built this old friend in several projects, but this time I built it on a two square inch board with surface mount components.

Photo1. The replacement circuit board has five switch pads. One yellow conductive button is shown mounted. This yellow casing acts like a spring for the conductive disk inside. The circuit components are soldered to the surface of the other side of the board.

Photo2.This is the complete system in action. The portable DVD TV display is turned sideways to match the rotation of the arcade monitor in the real system.

Listing1. These are the first eight entries in the 8080 instruction-decode table. Each four-byte entry contains three jump vectors (for GET, OPERATION, and SET) plus two parameter fields. The complete table consumes 1K bytes of shared memory.

dispatch\_table ' 29 26 18 9 0

' yyy\_xxx\_cccccccc\_bbbbbbbbb\_aaaaaaaaa

'

' VECTOR-1 (a) VECTOR-2 (b) VECTOR-3 (c) PARAM-V1 (x) PARAM-V3 (y) OP ASM FLAGS

long (nop\_v1<<Dv1) + (nop\_v2<<Dv2) + (nop\_v3<<Dv3) ' 00 NOP --------

long (get\_code\_word<<Dv1) + (nop\_v2<<Dv2) + (setR16<<Dv3) + (dec\_b<<Dr3) ' 01wlwm LD BC,w --------

long (getR8<<Dv1) + (nop\_v2<<Dv2) + (setBR16<<Dv3) + (dec\_a<<Dr1) + (dec\_b<<Dr3) ' 02 LD (BC),A --------

long (getR16<<Dv1) + (opINC16<<Dv2) + (setR16<<Dv3) + (dec\_b<<Dr1) + (dec\_b<<Dr3) ' 03 INC BC --------

long (getR8<<Dv1) + (opINC<<Dv2) + (setR8<<Dv3) + (dec\_b<<Dr1) + (dec\_b<<Dr3) ' 04 INC B sz-h-v0-

long (getR8<<Dv1) + (opDEC<<Dv2) + (setR8<<Dv3) + (dec\_b<<Dr1) + (dec\_b<<Dr3) ' 05 DEC B sz-u-v1-

long (get\_code\_byte<<Dv1) + (nop\_v2<<Dv2) + (setR8<<Dv3) + (dec\_b<<Dr3) ' 06bb LD B,b --------

long (nop\_v1<<Dv1) + (opRLCA<<Dv2) + (nop\_v3<<Dv3) ' 07 RLCA ---0--0x

Listing2. Instruction are decoded in three steps: GET, OPERATION, and SET. An example of each is shown here. The GET combines the H and L registers into an address and reads the data from memory. This OPERRATION waits on the port-cog to return the I/O value. The SET uses one of the parameter field from the table to identify an 8-bit register.

get\_word\_hl

mov address, h\_reg ' H is the upper register of pair

shl address,#8 ' Shift it into place

or address, l\_reg ' OR in the lower byte

call #read\_memory\_word ' Read the word from 8080 ram

jmp #vect\_2 ' Done

ioIn wrlong data\_8,io\_port ' Write the port address

mov data\_8,#io\_cmd\_in ' Write the ...

wrlong data\_8,io\_command ' ... IN command

ioWI rdlong data\_8,io\_command wz ' Wait for port handler ...

if\_nz jmp #ioWI ' ... to respond

rdlong a\_reg,io\_data ' Get the value

jmp #vect\_3 ' Done

setR8 movd sr8a, paramRegD ' Set the address of the destination register

nop ' Required gap before access

sr8a mov 0,data\_8 ' Copy data\_8 to target register

jmp #fetch ' Done

Figure2. The COGs communicate with one another through parameter blocks in shared memory. The 16K RAM/ROM of the emulated system lives in the propeller shared memory too.

currentRow

portCommand

interrupt

soundBits

sid registers

SI ROM (8K)

SI RAM (8K)

portAddress

portValue

Shared Memory

COG

SISound

SIDCOG

SIPorts

CPU\_8080

SIVideo

Disk\_hdw

COG