

```

1  --##### MCD #####
2  --# Company:      Eklektik Design
3  --# Engineer:     Micah Richards
4  --#
5  --# Create Date:   21:29 3/18/17
6  --# Module Name:   Monitor Program
7  --# Project Name:  MicroComputer Design
8  --# Target Devices: MCD Board
9  --# Tool versions:  Xilinx 14.7
10 --# Description:    Chip select controls
11 --#
12 --#####
13
14 --##### Libraries #####
15 library IEEE;
16 use IEEE.STD_LOGIC_1164.ALL;
17
18 --##### Entity #####
19 entity Main_Code is
20     Port (
21
22 --##### Ports #####
23         CPLD_CLK, CPLD_Button: in STD_LOGIC;
24         Read_Write, Add_Data_Valid, L_Data_Ready, H_Data_Ready: in STD_LOGIC;
25         ROM_H_Enable, ROM_H_Output, ROM_H_Input: out STD_LOGIC;
26         ROM_L_Enable, ROM_L_Output, ROM_L_Input: out STD_LOGIC;
27         RAM_H_Enable, RAM_H_Output, RAM_H_Input: out STD_LOGIC;
28         RAM_L_Enable, RAM_L_Output, RAM_L_Input: out STD_LOGIC;
29         DUART_Enable, DUART_Read_Write, DUART_Reset: out STD_LOGIC;
30         DUART_Data_ACK: in STD_LOGIC;
31         CPU_Clock, CPU_Data_ACK, CPU_Reset, CPU_Halt: out STD_LOGIC;
32         CPU_Valid_Periph_Add, CPU_Bus_Grant_ACK, CPU_Bus_ERR, CPU_Bus_REQ: out STD_Logic;
33         CPU_Read_Write, CPU_H_Data_Ready, CPU_L_Data_Ready, CPU_Enable, CPU_Add_Valid: in STD_LOGIC;
34         CPU_Bus_Grant, CPU_Valid_Memory_Add: in STD_Logic;
35         ADDR: in STD_LOGIC_VECTOR (3 downto 0);
36         LED: out STD_LOGIC_VECTOR (3 downto 0)
37     );
38 end Main_Code;
39
40 --##### Behavior #####
41 architecture Behavioral of Main_Code is
42
43 --##### Signals #####
44 signal I_Count1: integer range 0 to 4500 := 1; --### First integer for clock divider
45 signal I_Count2: integer range 0 to 25 := 1; --### Second integer for clock divider
46 signal I_CPU_Clock: STD_LOGIC := '0'; --### 7.5kHz clock for CPU
47 signal I_Reset_Triggered: STD_Logic:= '0'; --### Asynchronous Reset Signal
48
49 begin
50
51 --##### Chip Select #####
52 --# Purpose: Interprets the CPU signals to enable and disable the proper chips
53 --#
54 --#####
55 process (CPLD_CLK)
56 begin
57     if (rising_edge(CPLD_CLK)) then
58         LED(2)      <= '1';
59         LED(1)      <= '1';
60         ROM_H_Enable <= '1';
61         ROM_H_Output <= '1';
62         ROM_L_Enable <= '1';
63         ROM_L_Output <= '1';
64         RAM_H_Enable <= '1';
65         RAM_H_Input  <= '1';
66         RAM_H_Output <= '1';
67         RAM_L_Enable <= '1';
68         RAM_L_Input  <= '1';
69         RAM_L_Output <= '1';
70         DUART_Enable <= '1';
71         CPU_Data_ACK <= '1';
72
73         if (CPU_Read_Write = '1' and CPU_Add_Valid = '0' ) then --### 1 Reading
74             if (ADDR = "0000") then --### ROM Access
75                 if (CPU_H_Data_Ready = '0') then --### High
76                     ROM_H_Enable <= '0';
77                     ROM_H_Output <= '0';
78                     CPU_Data_ACK <= '0';
79                     LED(2) <= '0';
80                 end if;
81
82                 if (CPU_L_Data_Ready = '0') then --### Low
83                     ROM_L_Enable <= '0';
84                     ROM_L_Output <= '0';
85                     CPU_Data_ACK <= '0';
86                     LED(2) <= '0';
87                 end if;
88
89                 elsif (ADDR = "1111") then --### RAM Access
90                     if (CPU_H_Data_Ready = '0') then --### High
91                         RAM_H_Enable <= '0';
92                         RAM_H_Output <= '0';
93                         CPU_Data_ACK <= '0';
94                         LED(2) <= '0';
95                     end if;
96
97                     if (CPU_L_Data_Ready = '0') then --### Low
98                         RAM_L_Enable <= '0';
99                         RAM_L_Output <= '0';
100                        CPU_Data_ACK <= '0';
101                        LED(2) <= '0';
102                    end if;
103
104                     elsif (ADDR = "0011") then --### DUART Access
105                         if (CPU_H_Data_Ready = '0' or CPU_L_Data_Ready = '0') then

```

```

106         DUART_Enable <= '0';
107         LED(1) <= '0';
108         DUART_Read_Write <= '1';
109         CPU_Data_ACK <= DUART_Data_ACK;
110         LED(2) <= DUART_Data_ACK;
111     end if;
112 end if;
113 elsif (CPU_Read_Write = '0' and CPU_Add_Valid = '0') then --#### Writing
114     if (ADDR = "1111") then --#### RAM Access
115         if (CPU_H_Data_Ready = '0') then
116             RAM_H_Enable <= '0'; --#### High
117             RAM_H_Input <= '0';
118             CPU_Data_ACK <= '0';
119             LED(2) <= '0';
120         end if;
121
122         if (CPU_L_Data_Ready = '0') then --#### Low
123             RAM_L_Enable <= '0';
124             RAM_L_Input <= '0';
125             CPU_Data_ACK <= '0';
126             LED(2) <= '0';
127         end if;
128     elsif (ADDR = "0011") then --#### DUART Access
129         if (CPU_H_Data_Ready = '0' or CPU_L_Data_Ready = '0') then
130             DUART_Enable <= '0';
131             LED(1) <= '0';
132             DUART_Read_Write <= '0';
133             CPU_Data_ACK <= DUART_Data_ACK;
134             LED(2) <= DUART_Data_ACK;
135
136         end if;
137     end if;
138 end if;
139 end if;
140
141 end process;
142
143 --##### Clock Select #####
144 --# Purpose: Allows the user to select a clock speed for
145 --#         Function Select
146 --#
147 --#####
148 process (CPLD_CLK)
149 begin
150     if (rising_edge(CPLD_CLK)) then
151         I_Count1 <= I_Count1 + 1;
152         if (I_Count1 = 4500) then
153             I_Count1 <= 1;
154             I_Count2 <= I_Count2 + 1;
155             if (I_Count2 = 25) then --#### 500Hz Divide by 4500*25
156                 if (CPLD_Button = '0') then
157                     I_Reset_Triggered <= '1';
158                 else
159                     I_Reset_Triggered <= '0';
160                 end if;
161                 I_Count2 <= 1;
162             end if;
163         end if;
164     end if;
165 end process;
166
167 --##### Reset #####
168 --# Purpose: Allows the user to reset the board via the button
169 --#
170 --#####
171 process (I_Reset_Triggered)
172 begin
173     if (I_Reset_Triggered = '1') then
174         CPU_Halt <= '0';
175         CPU_Reset <= '0';
176         DUART_Reset <= '0';
177     else
178         CPU_Halt <= 'Z';
179         CPU_Reset <= 'Z';
180         DUART_Reset <= '1';
181     end if;
182 end process;
183
184 --##### Immediate Updates #####
185 CPU_Clock <= CPLD_CLK; --#### write 7.5 kHz Clock to CPU
186 LED(0) <= CPU_Read_Write;
187 LED(3) <= CPU_Add_Valid;
188
189
190 --##### Unused Pins Tied High #####
191 CPU_Bus_ERR <= '1';
192 CPU_Valid_Periph_Add <= '1';
193 CPU_Bus_Grant_ACK <= '1';
194 CPU_Bus_REQ <= '1';
195 end Behavioral;
196
197

```