**Coffee Pot Assignment 1 Code:**

**Aidan Johnson**

**Assignment1CoffeePot\_Sim533.cpp:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Assignment1CoffeePot\_Sim533.cpp

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**#include** <sys/platform.h>

**#include** "adi\_initialize.h"

**#include** "Assignment1CoffeePot\_Sim533.h"

/\*\*

\* If you want to use command program arguments, then place them in the following string.

\*/

**char** \_\_argv\_string[] = "";

**int** **main**(**int** argc, **char** \*argv[])

{

/\*\*

\* Initialize managed drivers and/or services that have been added to

\* the project.

\* @return zero on success

\*/

adi\_initComponents();

**#ifdef** \_\_ADSPBF533\_\_

**printf**("Starting the Coffee Pot Assignment\n");

startCoffeePot();

**#endif**

**#ifdef** \_\_ADSPBF609\_\_

printf("Starting the Coffee Pot Assignment\n");

startCoffeePot();

**#endif**

/\* Begin adding your custom code here \*/

return 0;

}

**Assignment1CoffeePot\_Sim533.h:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Assignment1CoffeePot\_Sim533.h

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**#ifndef** \_\_ASSIGNMENT1COFFEEPOT\_SIM533\_H\_\_

**#define** \_\_ASSIGNMENT1COFFEEPOT\_SIM533\_H\_\_

**#include** "../../ENCM511\_SpecificFiles/ENCM511\_include/CoffeePot\_SimulatorFunctions2017.h"

**#include** <stdio.h>

//Prototypes for Coffee Pot Assignment 1 Functions

**void** **startCoffeePot**(**void**);

**void** **showNameOfProcessorUsed**(**void**);

**void** **DemonstrateLEDControl\_CPP**(**void**);

**void** **initializingCoffeePot**(COFFEEPOT\_DEVICE\* coffeePot[]);

**void** **activateLEDControl**(COFFEEPOT\_DEVICE \*coffeePot[]);

**void** **activateWaterControl**(COFFEEPOT\_DEVICE \*coffeePot[]);

**void** **activateHeatControl**(COFFEEPOT\_DEVICE \*coffeePot[]);

**void** **LEDControlDemo**(COFFEEPOT\_DEVICE \*coffeePot);

**void** **fillCoffeePotToWaterLEvel**(COFFEEPOT\_DEVICE \*coffeePot);

**void** **heatWaterToTemperature**(COFFEEPOT\_DEVICE \*coffeePot);

**void** **checkForCoffeePod**(COFFEEPOT\_DEVICE \*coffeePot);

//Prototypes for Coffee Pot Assignment 1 Assembly Functions

extern "C" **void** **demonstrateLEDASM**(COFFEEPOT\_DEVICE \*coffeePot);

extern "C" **void** **checkForCoffeePodASM**(COFFEEPOT\_DEVICE \*coffeePot);

**#endif** /\* \_\_ASSIGNMENT1COFFEEPOT\_SIM533\_H\_\_ \*/

CheckForCoffeePodASM.asm:

/\*

\* checkForCoffeePodASM.asm

\*

\* Created on: Oct 30, 2019

\* Author: aidan

\*/

**#include** <blackfin.h>

**.section** L1\_data;

**.section** program;

**#define** INCOMING\_PAR R0

**#define** TEMPERATURE 95

**#define** COFFEEPOD 0x0800(Z)

**.global** \_checkForCoffeePodASM;

**\_checkForCoffeePodASM:**

[--SP] = R4;

[--SP] = P4;

LINK 16;

P4 = INCOMING\_PAR;

R4 = W[P4](Z);

R0 = P4;

**.extern** \_\_Z22CurrentTemperature\_CPPP16COFFEEPOT\_DEVICE;

CALL \_\_Z22CurrentTemperature\_CPPP16COFFEEPOT\_DEVICE;

R2 = R0;

R3 = TEMPERATURE;

CC = R3 <= R2;

If CC jump second;

jump Out;

**second:**

R1 = COFFEEPOD;

R2 = R4 & R1;

CC = R2 == 0;

If CC jump coffeePod;

**coffeePod:**

R1 = COFFEEPOD;

R0 = R4 | R1;

[P4] = R0;

**Out:**

UNLINK;

[SP++] = P4;

[SP++] = R4;

**\_checkForCoffeePodASM.end:**

RTS;

**demonstrateLEDASM.asm:**

/\*

\* demonstrateLEDASM.asm

\*

\* Created on: Oct 30, 2019

\* Author: aidan

\*/

**#include** <blackfin.h>

**#define** MASK\_LED 0xf000(Z)

**#define** INCOMING\_PAR R0

**#define** LED1BIT 0x1000(Z)

**#define** LED2BIT 0x2000(Z)

**#define** LED3BIT 0x4000(Z)

**#define** LED4BIT 0x8000(Z)

**#define** MASK\_BITS11TO0 0x0fff(Z)

**.section** L1\_data;

**.section** program;

**.global** \_demonstrateLEDASM;

**\_demonstrateLEDASM:**

LINK 20;

P0 = INCOMING\_PAR;

nop;

nop;

nop;

nop;

R1 = W[P0](Z);

R0 = MASK\_LED;

R1 = R1 & R0;

R1 = R1 >> 12;

CC = R1 == 0;

IF CC JUMP LED1;

CC = R1 == 1;

IF CC Jump LED2;

CC = R1 == 2;

IF CC Jump LED3;

R3 = 4;

CC = R1 == R3;

IF CC Jump LED4;

R3 = 8;

CC = R1 == R3;

IF CC Jump LED1;

**LED1:**

R2 = LED1BIT;

R1 = [P0];

R3 = MASK\_BITS11TO0;

R1 = R1 & R3;

R1 = R1 | R2;

[P0] = R1;

jump out;

**LED2:**

R2 = LED2BIT;

R1 = [P0];

R3 = MASK\_BITS11TO0;

R1 = R1 & R3;

R1 = R1 | R2;

[P0] = R1;

jump out;

**LED3:**

R2 = LED3BIT;

R1 = [P0];

R3 = MASK\_BITS11TO0;

R1 = R1 & R3;

R1 = R1 | R2;

[P0] = R1;

jump out;

**LED4:**

R2 = LED4BIT;

R1 = [P0];

R3 = MASK\_BITS11TO0;

R1 = R1 & R3;

R1 = R1 | R2;

[P0] = R1;

jump out;

**out:**

UNLINK;

**\_demonstrateLEDASM.end:**

RTS;

MainCoffeePot.cpp:

/\*

\* MainCoffeePot.cpp

\*

\* Created on: Oct 23, 2019

\* Author: aidan

\*/

**#include** "Assignment1CoffeePot\_Sim533.h"

**#define** MAX\_WATER\_LEVEL 600

**#define** MAX\_TEMPERATURE 100

**#define** NUMBEROFCOFFEEPOTS 4

**#define** SHOW\_FUNCTION\_STUB\_INFORMATION 1

**#define** MASK\_BITS\_15TO2\_AND\_BIT0 0xfffd

**#define** MASK\_LED 0xf000

**#define** MASK\_BITS\_15TO3\_AND\_BITS1TO0 0xfffb

**#define** MASK\_BITS\_15TO4\_AND\_BITS2TO0 0xfff7

**#define** WATERINFLOW 30

**#define** FULLHEATBOOST 15

**#define** HEATVALUE 190

**#define** TRUE 1

**#define** ZERO 0

**#define** TEMPERATURETOINSERTCOFFEEPOD 95

//ENUM for Coffeepot\_ID to make it an array

COFFEEPOT\_ID COFFEEPOTIDS[] = {*COFFEEPOT1*, *COFFEEPOT2*, *COFFEEPOT3*, *COFFEEPOT4*};

**void** **startCoffeePot**(**void**)

{

showNameOfProcessorUsed();

Init\_CoffeePotSimulation(NUMBEROFCOFFEEPOTS, *USE\_TEXT\_AND\_GRAPHICS\_GUIS*); //Initializing the simulation

**char** uniqueCoffeePot1to3Name[] = "Aidan";

**char** uniqueCoffeePot4Name[] = "Aidanj";

COFFEEPOT\_DEVICE\* coffeePotsBaseAddress[NUMBEROFCOFFEEPOTS];

for (**int** j = ZERO; j < (NUMBEROFCOFFEEPOTS - 1); j++)

{

coffeePotsBaseAddress[j] = Add\_CoffeePotToSystem\_PlugAndPlay(COFFEEPOTIDS[j], uniqueCoffeePot1to3Name);

}

coffeePotsBaseAddress[3] = Add\_CoffeePotToSystem\_PlugAndPlay(COFFEEPOTIDS[3], uniqueCoffeePot4Name); //Making the fourth coffeepot have a different capacity

initializingCoffeePot(coffeePotsBaseAddress);

activateLEDControl(coffeePotsBaseAddress);

activateWaterControl(coffeePotsBaseAddress);

activateHeatControl(coffeePotsBaseAddress);

while(TRUE)

{

for(**int** j = ZERO; j < NUMBEROFCOFFEEPOTS; j++)

{

//LEDControlDemo(coffeePotsBaseAddress);

demonstrateLEDASM(coffeePotsBaseAddress[j]); //Using the ASM function instead

fillCoffeePotToWaterLEvel(coffeePotsBaseAddress[j]);

heatWaterToTemperature(coffeePotsBaseAddress[j]);

checkForCoffeePodASM(coffeePotsBaseAddress[j]); //Using the ASM function instead

//checkForCoffeePod(coffeePotsBaseAddress[j]);

}

FastForward\_OneSimulationTIC(\*coffeePotsBaseAddress);

}

}

inline **void** **Show\_Function\_Stub\_Information**(**char** \*functionNameInformation)

{

if(SHOW\_FUNCTION\_STUB\_INFORMATION)

**printf**("%s \n", functionNameInformation);

}

**void** **showNameOfProcessorUsed**(**void**) //This function is displaying what processor is being used

{

**#if** defined(\_\_ADSPBF533\_\_)

**char** processor[] = "\_\_ADSPBF533\_\_";

**#else**

**char** processor[] = "\_\_ADSOBF609\_\_";

**#endif**

**printf**("Aidan's Coffee pots are running on %s system \n", processor);

}

**void** **initializingCoffeePot**(COFFEEPOT\_DEVICE \*coffeePot[]) //This function will be activating the 4 different coffeepots

{

**unsigned** **short** **int** currentControlRegister; //declaring a currentControlRegister variable

for(**int** j = ZERO; j < NUMBEROFCOFFEEPOTS; j++)

{

currentControlRegister = ReadControlRegister\_CPP(coffeePot[j]);

**unsigned** **short** **int** newControlRegister = INITandSTAYPOWEREDON\_BIT;

coffeePot[j] -> controlRegister = newControlRegister;

}

for(**int** j = ZERO; j < NUMBEROFCOFFEEPOTS; j++)

{

while((currentControlRegister & DEVICE\_READY\_BIT\_RO)!= DEVICE\_READY\_BIT\_RO)

{

FastForward\_OneSimulationTIC(coffeePot[j]);

currentControlRegister = ReadControlRegister\_CPP(coffeePot[j]);

}

}

}

**void** **activateLEDControl**(COFFEEPOT\_DEVICE \*coffeePot[])

{

**unsigned** **short** **int** currentControlRegister;

for(**int** j = ZERO; j < NUMBEROFCOFFEEPOTS; j++)

{

currentControlRegister = coffeePot[j] -> controlRegister;

currentControlRegister = currentControlRegister & MASK\_BITS\_15TO2\_AND\_BIT0;

currentControlRegister = currentControlRegister | LED\_DISPLAY\_ENABLE\_BIT;

coffeePot[j] -> controlRegister = currentControlRegister;

}

FastForward\_OneSimulationTIC(\*coffeePot);

}

**void** **activateWaterControl**(COFFEEPOT\_DEVICE \*coffeePot[])

{

**unsigned** **short** **int** currentControlRegister;

for (**int** j = ZERO; j < NUMBEROFCOFFEEPOTS; j++)

{

currentControlRegister = coffeePot[j] -> controlRegister;

currentControlRegister = currentControlRegister & MASK\_BITS\_15TO3\_AND\_BITS1TO0;

currentControlRegister = currentControlRegister | WATER\_ENABLE\_BIT;

coffeePot[j] -> controlRegister = currentControlRegister;

}

FastForward\_OneSimulationTIC(\*coffeePot);

}

**void** **activateHeatControl**(COFFEEPOT\_DEVICE \*coffeePot[])

{

**unsigned** **short** **int** currentControlRegister;

for(**int** j = ZERO; j < NUMBEROFCOFFEEPOTS; j++)

{

currentControlRegister = coffeePot[j] -> controlRegister;

currentControlRegister = currentControlRegister & MASK\_BITS\_15TO4\_AND\_BITS2TO0;

currentControlRegister = currentControlRegister | HEATER\_ENABLE\_BIT;

coffeePot[j] -> controlRegister = currentControlRegister;

}

FastForward\_OneSimulationTIC(\*coffeePot);

}

**void** **LEDControlDemo**(COFFEEPOT\_DEVICE \*coffeePot)

{

**unsigned** **short** **int** currentControlRegister = coffeePot -> controlRegister;

**unsigned** **short** **int** currentLEDState;

currentLEDState = (currentControlRegister & MASK\_LED) >> 12;

switch(currentLEDState)

{

case 0:

currentControlRegister = (currentControlRegister & ~MASK\_LED) | LED1\_BIT;

break;

case 1:

currentControlRegister = (currentControlRegister & ~MASK\_LED) | LED2\_BIT;

break;

case 2:

currentControlRegister = (currentControlRegister & ~MASK\_LED) | LED3\_BIT;

break;

case 4:

currentControlRegister = (currentControlRegister & ~MASK\_LED) | LED4\_BIT;

break;

case 8:

currentControlRegister = (currentControlRegister & ~MASK\_LED) | LED1\_BIT;

break;

}

coffeePot -> controlRegister = currentControlRegister;

}

**void** **fillCoffeePotToWaterLEvel**(COFFEEPOT\_DEVICE \*coffeePot)

{

**unsigned** **short** **int** currentControlRegister = coffeePot -> controlRegister;

**unsigned** **short** **int** currentWaterLevel = CurrentWaterLevel\_CPP(coffeePot);

if (currentWaterLevel < MAX\_WATER\_LEVEL - (MAX\_WATER\_LEVEL/8))

{

coffeePot -> waterInFlowRegister = WATERINFLOW;

currentControlRegister = currentControlRegister & MASK\_BITS\_15TO3\_AND\_BITS1TO0;

currentControlRegister = currentControlRegister | WATER\_ENABLE\_BIT;

coffeePot -> controlRegister = currentControlRegister;

}

else

{

coffeePot -> waterInFlowRegister = ZERO;

currentControlRegister = currentControlRegister & ~WATER\_ENABLE\_BIT;

coffeePot -> controlRegister = currentControlRegister;

}

}

**void** **heatWaterToTemperature**(COFFEEPOT\_DEVICE \*coffeePot)

{

**unsigned** **short** **int** currentControlRegister = coffeePot -> controlRegister;

**unsigned** **short** **int** currentTemperature = CurrentTemperature\_CPP(coffeePot);

if(currentTemperature < MAX\_TEMPERATURE)

{

coffeePot -> heaterRegister = HEATVALUE;

coffeePot -> heaterBoostRegister = FULLHEATBOOST;

currentControlRegister = currentControlRegister & MASK\_BITS\_15TO4\_AND\_BITS2TO0;

currentControlRegister = currentControlRegister | HEATER\_ENABLE\_BIT;

coffeePot -> controlRegister = currentControlRegister;

}

else

{

coffeePot -> heaterRegister = ZERO;

coffeePot -> heaterBoostRegister = ZERO;

currentControlRegister = currentControlRegister & ~HEATER\_ENABLE\_BIT;

coffeePot -> controlRegister = currentControlRegister;

}

}

**void** **checkForCoffeePod**(COFFEEPOT\_DEVICE \*coffeePot)

{

**unsigned** **short** **int** currentControlRegister = coffeePot -> controlRegister;

**unsigned** **short** **int** currentTemperature = CurrentTemperature\_CPP(coffeePot);

if (currentTemperature >= TEMPERATURETOINSERTCOFFEEPOD)

{

if((currentControlRegister & COFFEEPOT\_INSERTED) == ZERO)

{

currentControlRegister = (currentControlRegister | COFFEEPOT\_INSERTED);

coffeePot -> controlRegister = currentControlRegister;

}

}

}