Project 3

Alberto Valdiviez, John Nguyen, Ryan Young

**ABSTRACT**

**INTRODUCTION**

**DISCUSSION OF THE PROJECT**

**Design Specification**

**Software Implementation**.

**RESULTS**

**ERRORS**

**ANALYSIS OF PROBLEMS**

**TEST PLAN**

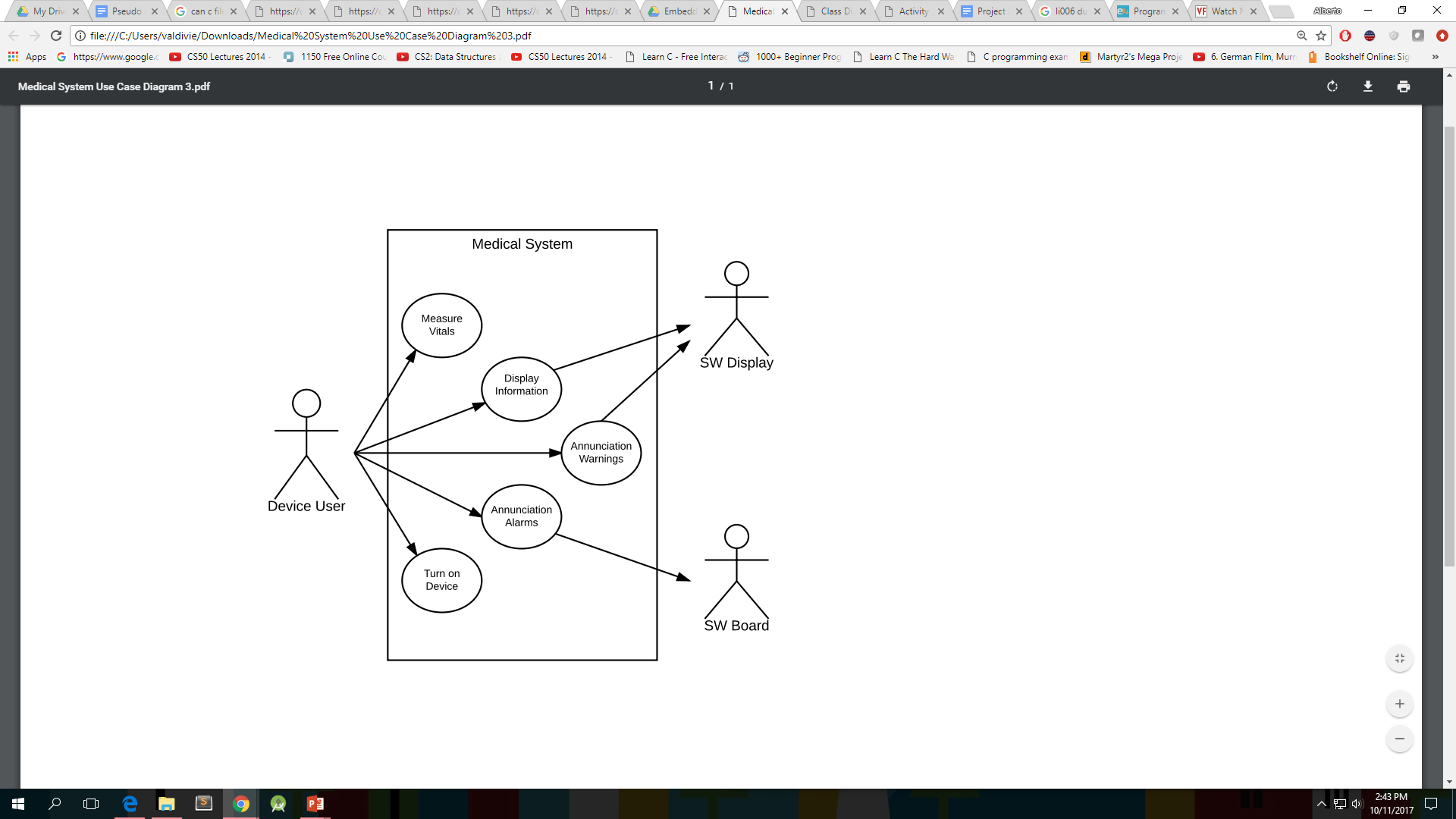
**TEST SPECIFICATION**

**SUMMARY**

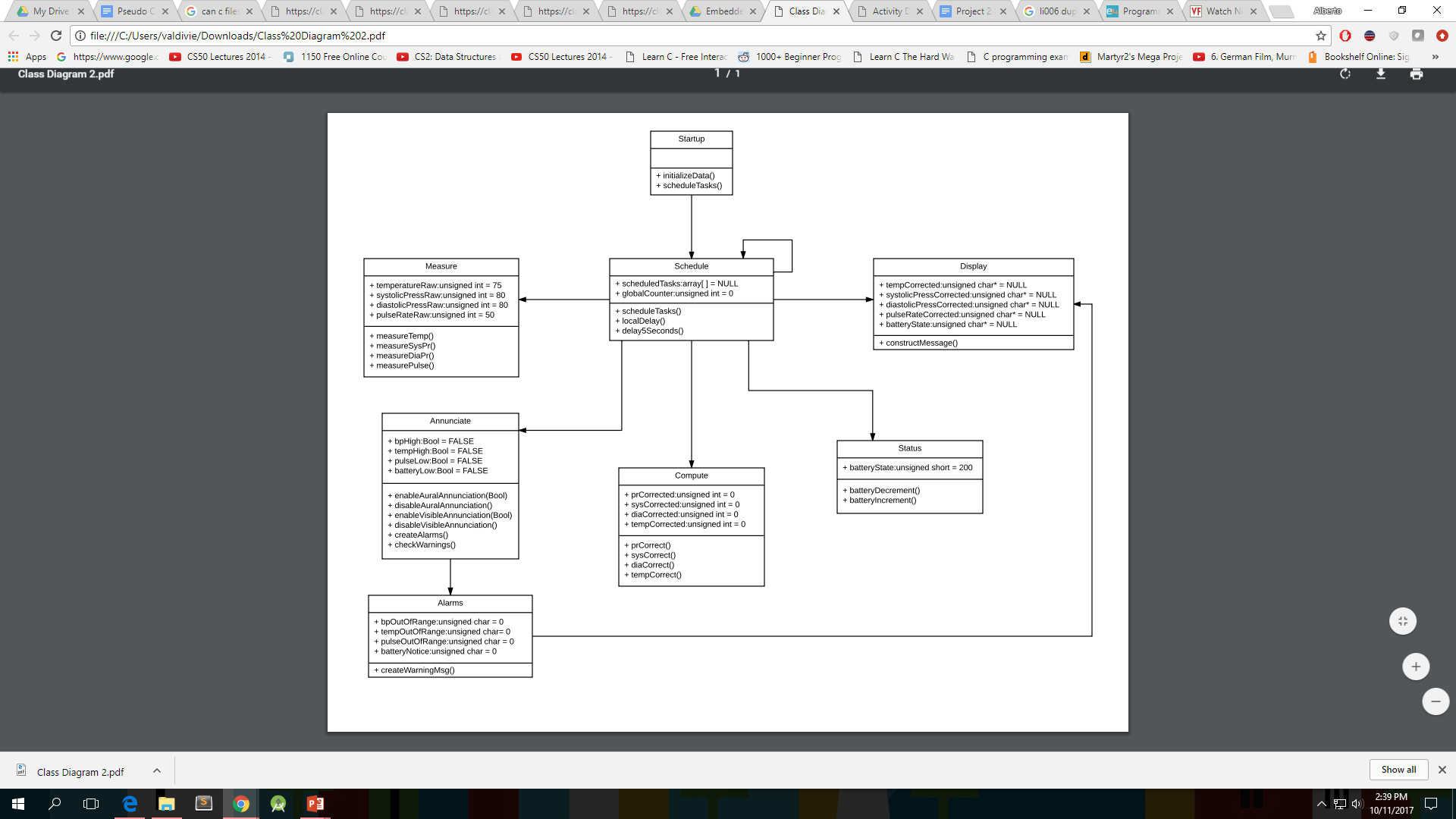
**CONCLUSION**

**APPENDIX**

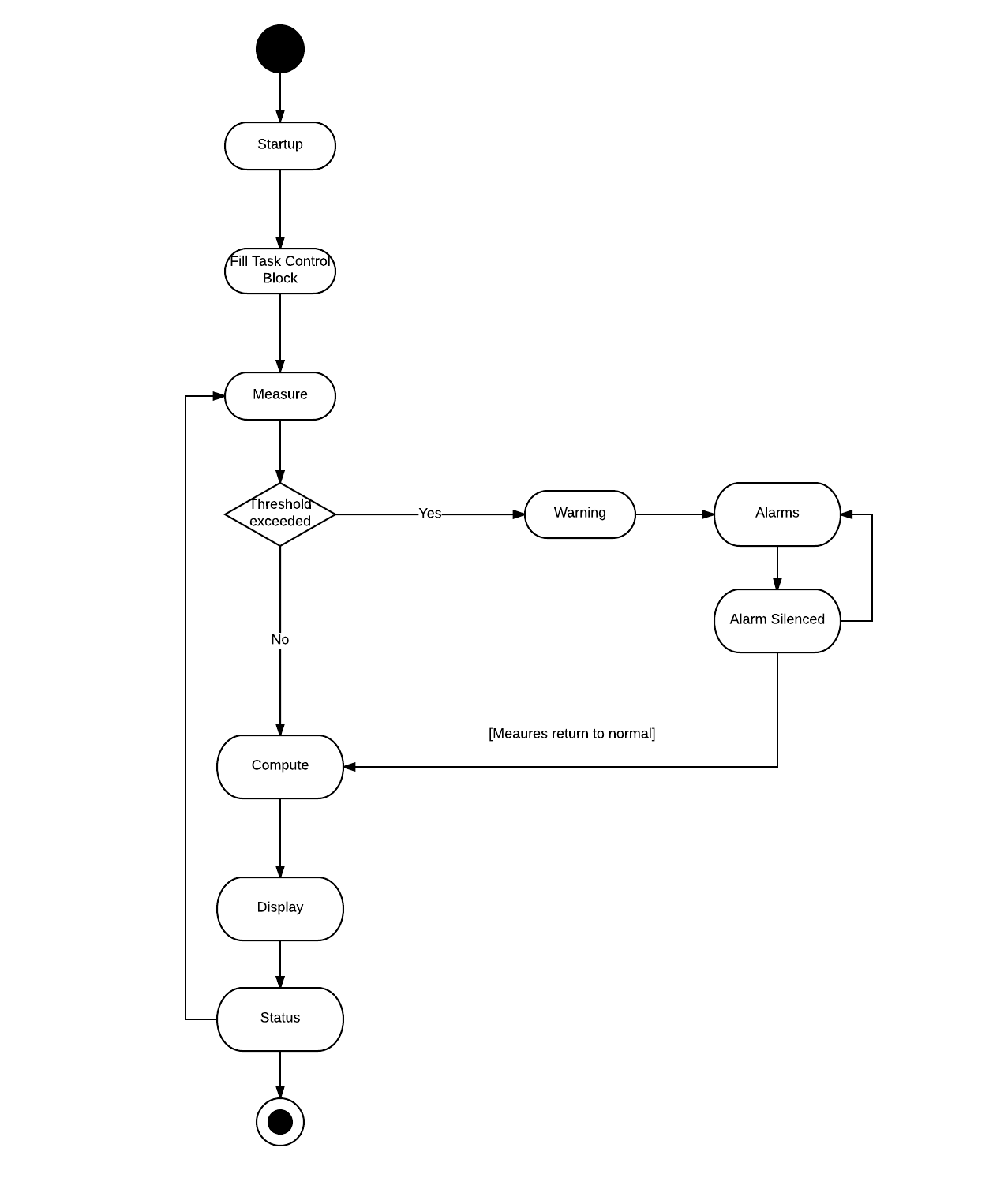
**Use Cases**

****

**Class Diagram**

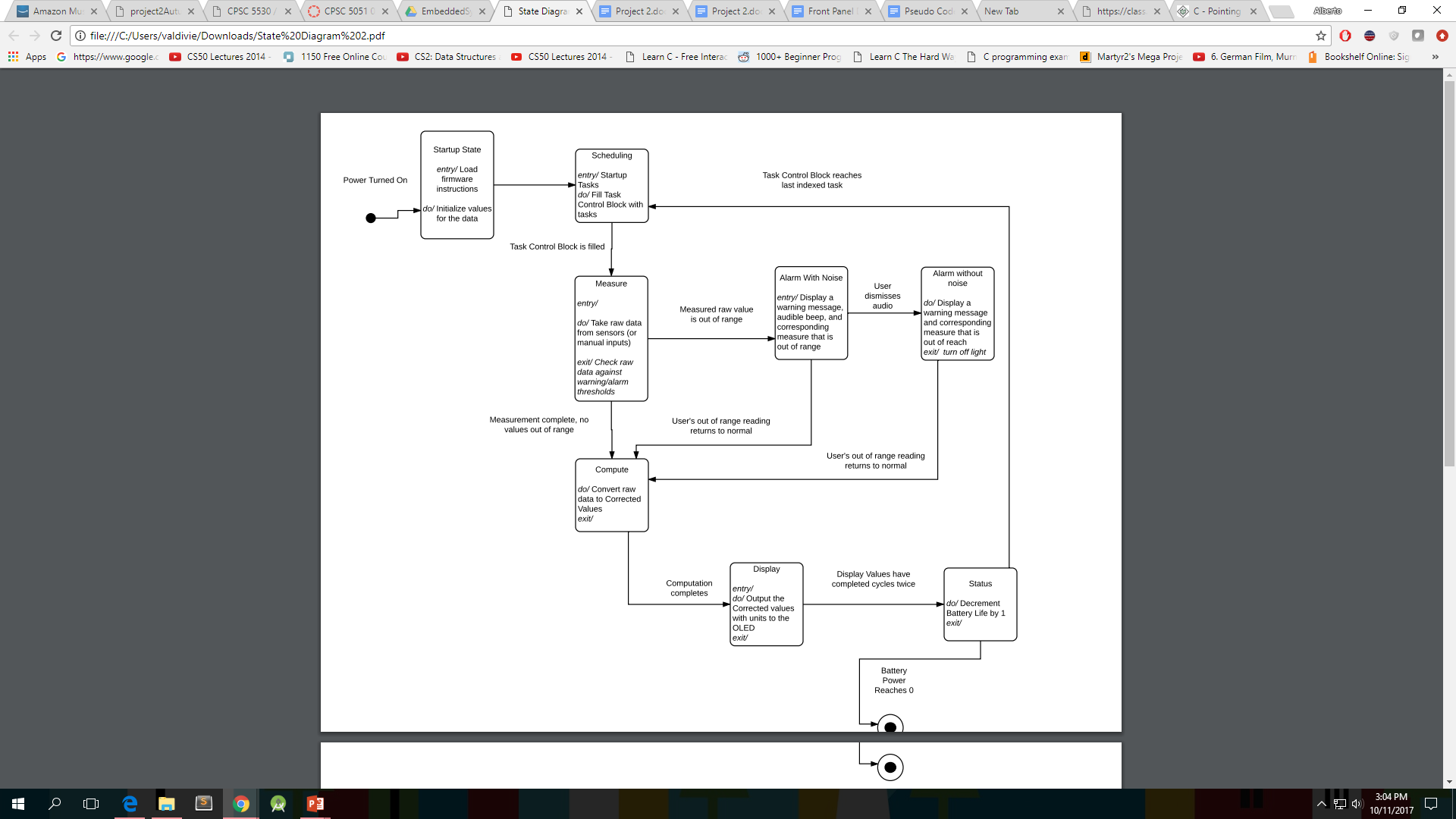
****

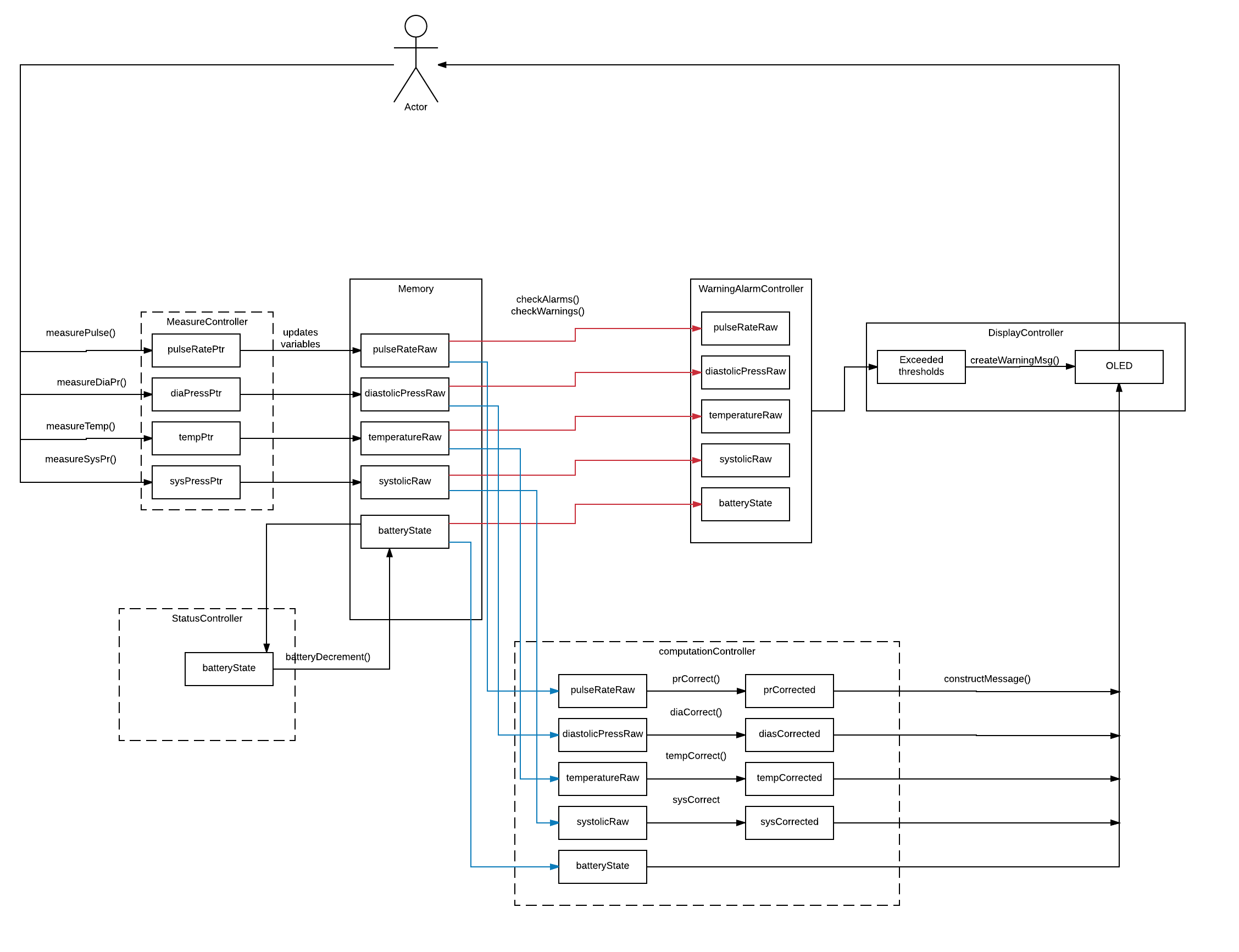
**Activity Diagram**

****

**Sequence Diagram**

**State Diagram**

****

**Module Block Diagram(High Level)**

**Function Prototypes - Pseudocode**

void compute(void\* data);  
 void measure(void\* data);  
 void stat(void\* data);  
 void alarm(void\* data);  
 void disp(void\* data);  
 void schedule(void\* data);

**Data Structures-Pseudocode**

struct measurements

{

unsigned int temperatureRaw;

unsigned int systolicPressRaw;

unsigned int diastolicPressRaw;

unsigned int pulseRateRaw;

}measurementsDefault={75,80,80,50};

struct display

{

unsigned char\* tempCorrected;

unsigned char\* systolicPressCorrected;

unsigned char\* diastolicPressCorrected;

unsigned char\* pulseRateCorrected;

}displayDefault={NULL,NULL,NULL,NULL};

struct status

{

static unsigned char batteryState;

}stateDefault{200};

struct alarms

{

unsigned char bpOutOfRange;

unsigned char tempOutOfRange;

unsigned char pulseOutOfRange;

}statusDefault{0,0,0};

struct warning

{

bool bpHigh;

bool tempHigh;

bool pulseLow;

}warningDefault{FALSE,FALSE,FALSE};

struct scheduler{

unsigned int globalCounter;

}schedulerDefault{0};

/\*\*\*\*\*

For Separate code snippet to be included in. compiler doesn’t recognize boolean type

enum \_myBool { FALSE = 0, TRUE = 1 };

typedef enum \_myBool Bool;

**\*\*\*/**

**Data Structrues Pointers -Pseudocode**

struct measureData //Holds pointers to the variables:

{

int\* temperatureRawPtr;

int\* systolicPressRawPtr;

int\* diastolicPressRawPtr;

int\* pulseRateRawPtr;

}

struct computeData

{

int\* temperatureRawPtr;

int\* systolicPressRawPtr;

int\* diastolicPressRawPtr;

int\* pulseRateRawPtr;

int\* tempCorrectedPtr;

int\* sysPressCorrectedPtr;

int\* diasCorrectedPtr;

int\* prCorrectedPtr;

}

struct displayData

{

char\* tempCorrectedPtr;

char\* sysCorrectedPtr;

char\* diasCorecctedPtr;

char\* prCorrectedPtr;

char\* batteryStatePtr;

}

struct warningAlarmData

{

int\* temperatureRawPtr;

int\* systolicPressRawPtr;

int\* diastolicPressRawPtr;

int\* pulseRateRawPtr;

int\* batteryStatePtr;

}

struct statusData

{

short\* batteryStatePtr;

}

struct schedulerData

{

int\* globalCounterPtr

}

**Compute Task-Pseudocode**

void computeTask(void \*data) {

// Recast task argument pointer to task’s data structure type

TaskData computeData;

void\* computeData = (void \*) &data;

// Dereference pointer

\*(TaskData\*) computeData;

// Perform transformations of raw values to corrected values

tempCorrect();

sysCorrect();

diaCorrect();

prCorrect();

}

tempCorrect(TaskData data) {

\*data.tempCorrected = 5 + (0.76 \* temperatureRaw);

}

sysCorrect(TaskData data) {

\*data.sysCorrected = 9 + (2 \* systolicRaw);

}

diaCorrect(TaskData data) {

\*data.diaCorrected = 6 + (1.5 \* diastolicRaw);

}

prCorrect(TaskData data) {

\*data.prCorrected = 8 + (3 \* pulseRateRaw);

}

**Scheduler Task -Pseudocode**

**Display Task Pseudocode**

**Alarm / Warning Task**

void warningAlarmTask(void \*data) {

// Recast task argument pointer to task’s data structure type

TaskData warningAlarmData;

void\* warningAlarmData = (void \*) &data;

// Dereference pointer

\*(TaskData\*) warningAlarmData;

// Check if any vitals are out of range and set variable to true if needed

checkWarnings() {

If (temperature out of range) {

Set temp bool to TRUE

} else if (blood pressure out of range) {

Set blood pressure bool to TRUE

} else if (pulse rate out of range) {

Set pulse rate bool to TRUE

} else if (battery life out of range) {

Set battery life bool to TRUE

}

}

// Check if any bools are true

If (!temp & !bp & !pulse & !battery) {

// Check if any annunciations are enabled

If (annunciation) {

// Disable annunciations

disableAuralAnnunciation()

disableVisibleAnnunication()

return;

} else {

return;

}

// Enable annunciations

} else {

enableAuralAnnunciation();

enableVisibleAnnunication();

createWarningMessages();

return;

}

}

**Status Task - Pseudocode**

void statusTask(void \*data) {

// Recast task argument pointer to task’s data structure type

TaskData statusData;

void\* statusData = (void \*) &data;

// Dereference pointer

\*(TaskData\*) statusData;

// Decrement battery state

decrementBattery(&statusData);

}

decrementBattery(TaskData \*statusData) {

\*statusData.batteryState++;

}

**Measure Task Pseudocode**

void measureTask(void\* measureData){

MeasureData\* measureDataPtr = (MeasureData\*) measureData;

// each measurement should pass the measureData by reference

//needs to count every time that the measurement task is entered as this value will be   
 used to automate measurements

incrementCounter(){

countMeasures ++;

}

void measureTemp(void\* measureDataPtr ){;

Check temperatureDirection and store;

if(temperature Raw>50 and direction = Increasing){

Direction = decreasing;

}

else if(temperature < 15 and direction = decreasing){

Direction = increasing;

}

Temperature Raw += countMeasures % 2 multiplied by direction

};

void measureSysBp(void\* measureDataPtr ){

If (diastoliccomplete and systolic > 100){

sysRaw= 80;

diastolicComplete = 0;

}

If (systolicPress<100){

If (countMeasure % 2 =0){

Increment on even counts by 3

}

else

Decrement on odd counts by 1

}

If(systolicPress>100)

systolicComplete=1

};

void measureDiaBp(void\* measureDataPtr ){

If (diastoliccomplete and diastolicPressure< 40){

sysRaw= 80;

diastolicComplete = 0;

}

If (systolicPress<100){

If (countMeasure % 2 =0){

Increment on even counts by 3

}

else

Decrement on odd counts by 1

}

If(systolicPress>100)

systolicComplete=1

};

void measurePR(void\* measureDataPtr ){

//If pulse rate is above 40 and increasing swap the direction  
 if (40<pulseRate && direction==Increasing){  
 direction = decreasing;  
 }  
 //If pulseRate is below 15 and decreasing swap the direction  
 else if (15>= pulseRate && direction==decreasing){   
 \*direction = increasing;  
 }  
 // increment or decrement (using the direction value)  
 if (countMeasure is even){  
 pulserate = pulseRate - direction;  
 }  
 //odd case  
 else {  
 pulseRate += direction \* 3  
 }

};

**Front Panel Diagram**

|  |
| --- |
| Systolic BP: 100 Diastolic BP: 100  Temp: 98.2 Pulse Rate: 89 Battery: 100%  WARNING: Temperature High  WARNING: Pulse Rate Low  WARNING: Blood Pressure High  WARNING: Battery Low |

**Citation**

* [**http://oxconltd.com/SeattleU/assignments/project2Autumn-17.pdf**](http://oxconltd.com/SeattleU/assignments/project2Autumn-17.pdf)
* EK-LM3S8962 Firmware Development Package User Guide
* [**https://class.ee.washington.edu/474/peckol/code/C/kernel1.c**](https://class.ee.washington.edu/474/peckol/code/C/kernel1.c)
* blinky.c from the stellarisware example code

**Contribution Statements**

Alberto Valdiviez

John Nguyen

Ryan Young