**UNIVERSIDAD TECNOLÓGICA DE QUERÉTARO**



SOFTWARE DEVELOPMENT PLAN

FOR THE

DC MOTOR SPEED CONTROLLER

Embedded Software Course

Ugarte, Gabriela

Chavez, Jose Luis

31/10/2019

## RECORD OF CHANGES

\***A-** ADDED **M**-MODIFIED **D**-DELETED

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **VERSION NUMBER** | **DATE** | **NUMBER OF FIGURE, TABLE, OR PARAGRAPH** | **A \***  **M**  **D** | **TITLE OR BRIEF DESCRIPTION** | **Change Request Number** |
| 1 | 14/09/2019 |  | **A**-Project Scope, deliverables, development methodology, Estimates, Planning Section |  |  |
| 2 | 28/09/2019 |  |  | Initial Revision |  |
| 3 | 12/10/2019 |  | **M-**Development methodology, Planning, Deliverables |  |  |
| 4 | 26/10/2019 |  | **A** – Design, Testing, Release  **M**- Deliverables, Front Page |  |  |
| 5 | 28/10/2019 |  |  | Second Revision |  |

Table of Contents

[RECORD OF CHANGES 2](#_Toc22985276)

[PROJECT SCOPE 4](#_Toc22985277)

[Document Overview 4](#_Toc22985278)

[System Overview 4](#_Toc22985279)

[Referenced Documents 4](#_Toc22985280)

[Overview of required work 5](#_Toc22985281)

[DELIVERABLES 5](#_Toc22985282)

[Change control 5](#_Toc22985283)

[Documents 6](#_Toc22985284)

[Development Buils 6](#_Toc22985285)

[Source Control for Deliverables 7](#_Toc22985286)

[DEVELOPMENT METHODOLOGY 8](#_Toc22985287)

[SCRUM 8](#_Toc22985288)

[SCRUM board 8](#_Toc22985289)

[Roles and Responsibilities 9](#_Toc22985290)

[ESTIMATES 9](#_Toc22985291)

[PLANNING 12](#_Toc22985292)

[Roles and Responsibilities 12](#_Toc22985293)

[Gannt Diagram 12](#_Toc22985294)

[Solving Problem Strategy 12](#_Toc22985295)

[Design 12](#_Toc22985296)

[Standards 12](#_Toc22985297)

[Naming Conventions 13](#_Toc22985298)

[Testing 14](#_Toc22985299)

[Verification strategy (BlackBox Test) 14](#_Toc22985300)

[WhiteBox Strategy 15](#_Toc22985301)

[Cyclomatic Complexity Redundance Index 15](#_Toc22985302)

[RELEASE 15](#_Toc22985303)

[Integration Test Strategy 15](#_Toc22985304)

[Validation Testing / Functional Testing 16](#_Toc22985305)

[Throughput and Flash and RAM measurement 16](#_Toc22985306)

[Results 16](#_Toc22985307)

[Lessons Learned 16](#_Toc22985308)

# PROJECT SCOPE

This Software Development Plan (SDP) establishes the plan for software implementation and test for a DC Motor Speed Controller. The DC Motor Speed Controller is being developed under the direction of the Embedded Software Companies from Queretaro committee. Updates to this SDP will address future upgrades to the DC Motor Speed Controller.

## Document Overview

This SDP identifies applicable policies, requirements, and standards for the DC Motor Speed Controller software development. It defines schedules, organization, resources, and processes to be followed for all software activities necessary to accomplish the development.

## System Overview

The DC Motor Speed Controller involves the development of an embedded software application using the Renesas S7G2 Starter Kit SK R7FS7G27H3A01CFC and Renesas e2 studio (Eclipse based). This application shall control the speed of a DC motor with a squared signal input at a constant frequency.

The PWM frequency shall be in between a range of 100 Hz to 1 KHz. Once the PWM frequency is selected, this shall be constant, only the “duty cycle” will be modifiable.

A Hall-effect sensor will be placed at the shaft of the DC motor to measure the speed of the motor by counting the amount of pulses to achieve one full spin.

The higher the speed of the motor, the higher will be the amount of pulses from the hall-effect sensor, and vice versa.

The motor shall regulate the speed to the value set by the user.

Additionally, an LCD shall display the motor speed and the Set Point (both in RPM’s), as well as the percentage of the PWM duty cycle value.

## Referenced Documents

The documents listed below were either used to create this document or are referenced in it:

1. Software Requirements Document extract from ReqView is included in the <Project\_Path>\1)Requirements\SWRA\_20190914.docx
2. Renesas S7G2 Datasheet

# Overview of required work

|  |  |  |  |
| --- | --- | --- | --- |
| Program Strategy | Define All Requirements First? | Multiple Development Cycles | Field iterim Softwarwe? |
| Grand Design | Yes | No | No |
| Incremental (Preplanned Product Improvement) | Yes | Yes | Maybe |
| Evolutionary | No | Yes | Yes |

Table 1. Key Features of Three DoD Program Strategies

Selected methodology is the Incremental Program Strategy (Preplanned Product Improvement).

The DC Motor Speed Controller will apply and Incremental (Preplanned Product Improvement) strategy to develop and evolve the functional capabilities of the System Software. In this program strategy the requirements are defined and then the development of the software consists on a sequence of builds. Each sequence of builds adds more capabilities until the system is complete.

All issues concerning cost, schedule, and incremental build content must be negotiated with the Project Manager.

# DELIVERABLES

## Change control

The change control types that will be driven in the Project will be:

* Change Request
* PR

The Change Control Process is defined in the Figure 1

Figure 1. Change Control flow

### Documents

Table 2 describes the list of documents that will be delivered for this Project. Please note that all document location is realtive to the path where this document is placed (*<PROJECT\_PATH).*

|  |  |
| --- | --- |
| **Document Name** | **Brief Description** |
| Software Development Plan | Master File for the Development Plan |
| SWRA | Software Requirement Analysis Document |
| Software Standards | Detailed use of the software standards in the Project |
| Software Design Document | Contains system and software diagrams |
| Planning | Gantt Project Plan |
| FMEA | Risk management File |
| BlackBox Test | Contains detailed information about the BlackBox tests |
| WhiteBox Test | Contains detailed information about the BlackBox tests |
| BlackBox Test Results | Contains the results from the BlackBox Tests |
| WhiteBox Test Results | Contains the results from the WhiteBox Tests |
| Integration Testing | Contains the results from the System Tests |
| Meeting Minutes | Document that contains the meeting minutes between Product owner, developers, and Scrum Master. |
| Quality Audit | Document for the project’s evaulation |

Table 2. Documents Delivered

### Development Buils

The complete RENESAS Project can be found at Software Requirements Document extract from ReqView is included in the <Project\_Path>\ DC\_Motor\_Controller.zip

Although several files are contained inside the RENESAS project, only those manually modified are listed in the Table 3.

|  |  |
| --- | --- |
| **File Name** | **Short Description** |
| guiapp\_event\_handlers.c | Event handlers for actions related to the LCD display |
| main\_thread\_entry.c | Main function. Timers, Interruptions, PWM are initialized. |
| common.h | Name definitions used throughout the RENESAS project. |
| adc.c | Reading of potenciometer signal and converting it and sampling the readings. |
| adc.h | Definition of functions used for the adc implementation. |
| controller.c | Calculate and get PWM; PID controller implementation: sintonized variables (kp, ki and kd), calculation of error. |
| controller.h | Definition of functions used for the controller implementation. |
| diagnostics.c | Get Setpoint, short to battery, short to ground. |
| diagnostics.h | Definition of functions used for the diagnostics implementation. |
| sensor.c | Signal processing of the hall effect sensor. Get current speed |
| sensor.h | Definition of functions used for the sensor implementation. |

Tabla 3. Code Files Delivered

## Source Control for Deliverables

GIT will be used for Data Configuration Management and Product Configuration Management. Development, Engineer, and Production releases will be managed through the GITHub environment. (See Figure 2)

The Development builds will be delivered through a branch called *dev* integrated in a Main Branch called Master.

The Documents will be delivered through a branch called *Docs* integrated in a Main Branch called Master.

The master branch will contain all builds and documents delivered for the final release.

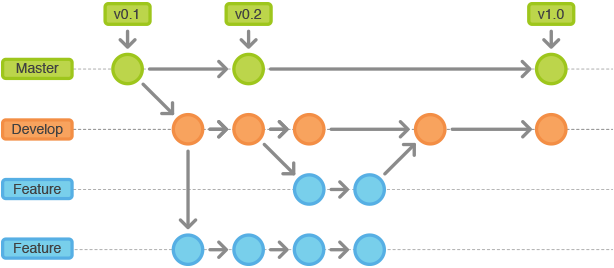


Figure 2. Workflow Cycle of Git. Image courtesy of <https://leanpub.com/>

# DEVELOPMENT METHODOLOGY

The DC Motor Speed Controller development will apply the following general methods:

1. The project will follow the defined processes documented in the SDP to conduct software requirements analysis and manage the Software Requirement Analysis (SWRA) Document. Express software requirements in a language that addresses an objective per statement and promotes measurable verification.
2. The project will adhere to the standards required by the SDP for design, coding and test methods for the development of the software.
3. The Software Development Group and the Software Test and Evaluation Group collaborate during software requirements analysis and then proceed with separate activities that are intended to ensure the software complies with the detailed software requirement specification.

## SCRUM

To predict risks, optimize delivering time, and solve problems along the project, the methodology used is SCRUM.

### SCRUM board

The SCRUM board was managed though ZenHub within GitHub. At ZenHub, all the activities defined for the project were monitored and tracked through their status in the board. See Figure 3.

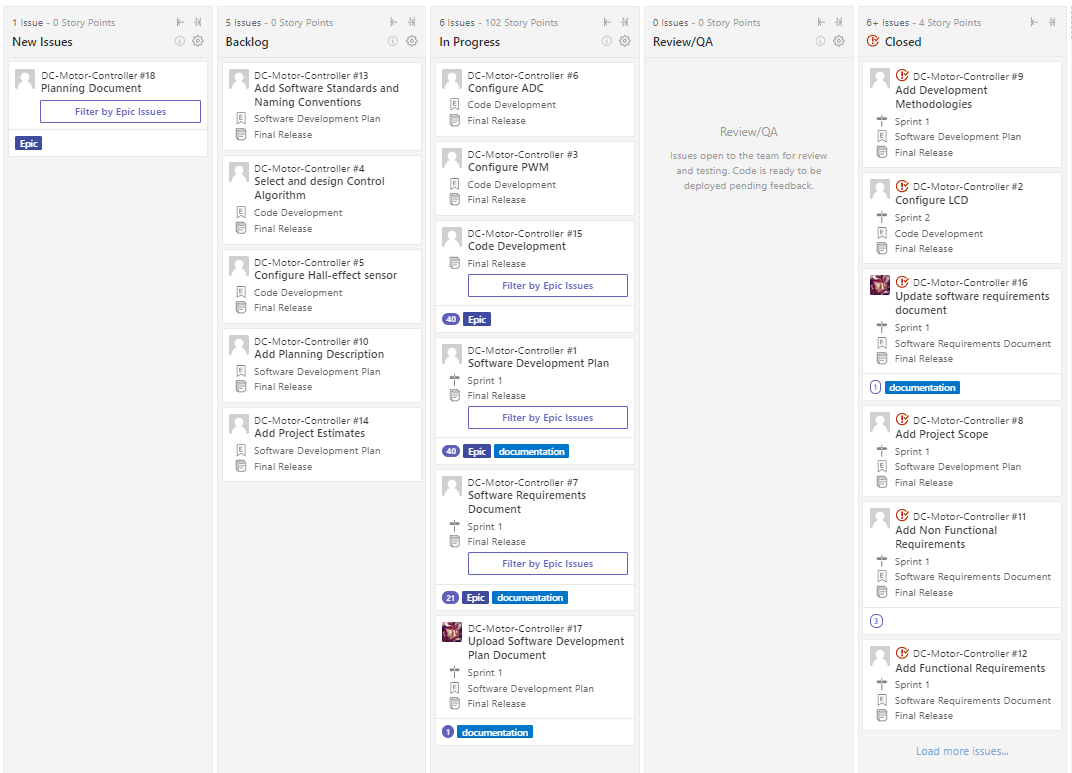


Figure 3. ZenHub Task Management

### Roles and Responsibilities

At the SCRUM methodology, it is needed to define the following roles:

1. Product Owner. Francisco Ávila (Tutor)
2. Scrum Master. Ugarte Gabriela, Chavez Jose Luis
3. Developers. Ugarte Gabriela, Chavez Jose Luis

The role of *Scrum Master* will be dynamic between both team members each Sprint.

Sprints will be carried out each week to keep track of activities and its progress to achieve the final release. The developers will share the progress of their assigned activities and extern any roadblock or dependency.

Additionally, a weekly meeting between the *Product Owner, Scrum master,* and *developers* will be held at the beginning of each sprint to review progress and overall doubts about the project.

The topics discussed in these meeting will be documented and uploaded to the <Project\_Path>\3) Planning/Meeting\_Minutes.docx

# ESTIMATES

To organize activities in such way that dependencies do not interfere with the deliverables and to identify potential risks at early stages of the plan, the following analysis was made to identify the necessities of the project.

The facts considered for the project are the following:

1. There is a workstation available with the following components:
   1. Test Bench
   2. PC
   3. Wireless Network (Wi-Fi) or wired network (Ethernet).
   4. Scope
   5. Multimeter
   6. Power Supply
   7. Tachometer
2. The following supplies are available:
3. RENESAS Development Board Model S7G2 SK R7FS7G27H3A01CFC
4. Display LCD 320X240, 2.4”
5. CESEQ\_P001
6. CESEQ\_C001
7. The following software is available:
8. Renesas e2 studio (Eclipse based)
9. GCCARM Compiler
10. GIT
11. Microsoft Office 2010
12. Matlab
13. The following human resources are available
    1. Product Owner
    2. Developers
    3. Scrum Master

The assumptions made for the project are the following:

1. Hardware is available and in good conditions
2. Software is available and has the necessary functionality to aid with the project’s deliverables.
3. Laboratory will be available to work on the project on Fridays from 02:00pm to 06:00pm and on Saturdays from 11:00 am to 1:00 pm
4. Team members will be available to work on the project on Fridays from 02:00pm to 06:00pm on Saturdays from 11:00 am to 1:00 pm
5. Project requirements will not change from those defined at the beginning of the course

The risks identified for the project are the described below:

1. Unable to work with RENESAS Board outside of UTEQ
2. Unaccurate planning due to lack of knowledge on project’s complexity.
3. Uncalibrated sensors
4. Unstable controller
5. Damaged components
6. Change in requirements
7. Team members’ unavailability to work on project outside of Fridays or Saturdays.

The risks identified in this project can be found at the <Project\_Path>\2. Design\FMEA.xlsx

Considering all the previous points, it was possible to make estimates for the activities that needed to be done in order to complete the project. Using the SCRUM methodology, the epics and stories shown in Table 4 below were defined.

|  |  |
| --- | --- |
| **Activity** | **Type** |
| **Software Development Plan** | **Epic** |
| Add Project Scope | Story |
| Add Deliverables | Story |
| Add Development Methodology | Story |
| Add Estimates | Story |
| Add Planning | Story |
| Add Problem Solving Stategy | Story |
| Add Design | Story |
| Add Testing | Story |
| Add Release | Story |
| Add Results and Lessons Learned | Story |
| **Planning Document** | **Epic** |
| Add Gantt Diagram | Story |
| Add Meeting Minutes | Story |
| **Software Design Document** | **Epic** |
| Add FMEA Document | Story |
| Add Software Standards | Story |
| Add software and system diagrams | Story |
| **Software Requirements Document** | **Epic** |
| Add functional requirements | Story |
| Add non-functional requirements | Story |
| **Code Development** | **Epic** |
| Configure LCD | Story |
| Configure PWM | Story |
| Configure Hall-effect sensor | Story |
| Select and design Control Algorithm | Story |
| Develop control algorithm | Story |
| Implement controller | Story |
| Test controller | Story |
| Short to Battery | Story |
| Short to Ground | Story |
| Memory corruption | Story |
| Button in short circuit | Story |
| **Integration** | **Epic** |
| Integration of modules | Story |
| **Software Tests** | **Epic** |
| Black Box Tests and Results | Story |
| White Box Tests and Results | Story |
| Integration Tests and Results | Story |
| Validation Tests and Results | Story |

Table 4. Activities planned at Task Management

# PLANNING

This SDP shall be maintained and modified to reflect the current plans, policies, processes, resources, and standards affecting the DC Motor Speed Controller Project.

Microsoft Excel will be used to develop and maintain the DC Motor Speed Controller Project master plan and schedule using a Gantt Diagram.

## Roles and Responsibilities

The roles and responsibilities for the project are listed below.

1. SCRUM Master. Maintains the master plan and schedule updated along with the Data Configuration Management and the Product Configuration Management.
2. Development Team. Manages the requirements data base, performs analysis to identify algorithms, high level data flow, interfaces and logical functions, and performs updates in response to PRs, performs analysis to create the functional model and design of the software system, and performs analysis to implement the test case and test procedures defined, plans, integrates, and execute tests.

## Gannt Diagram

The Master Schedule is stablished in the Gantt Diagram included at the <Project\_Path>\3) Planning\Planning.xlsx

# Solving Problem Strategy

The Failure Mode and Effects Analysis (FMEA) is an structured approach to discover potential failures that may exist within the design of a product or process. The sooner a failure is discover, the less it will cost to repair it. FMEA helps to create multiple chocies of mitigating risks, higher capability of Verification and Validation of changes, as well as lower cost solutions.

The FMEA for the project and the risks involved during the development of the software is included at the <Project\_Path>\2) Design/FMEA.xlsx

# Design

The Software Design Document is included at the <Project\_Path>\2) Design/SoftwareDesignDocument.docx

## Standards

The coding standard used for this project was **C99**, formally known as ISO/IEC 9899:2018.

In order to validate that the standard was followed correctly, an additional software tool was used, *Cppcheck v1.87*. See Figure 4.

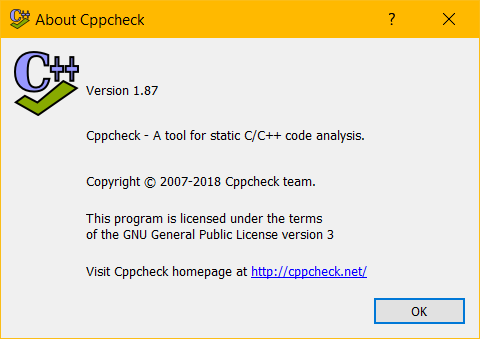


Figure 4. Cppcheck

This program allows users to analyze C code files with a specific standard from its catalogue; in this case, C99 was selected. See figure 5.

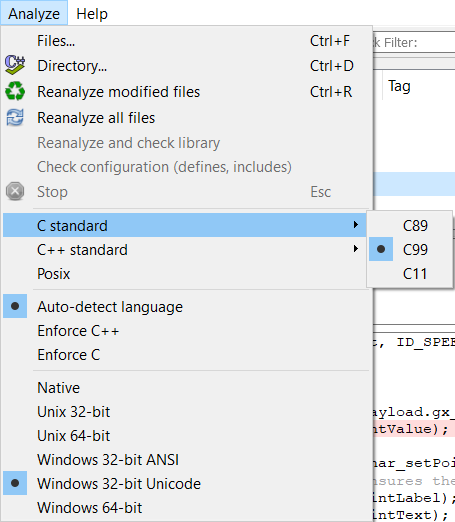


Figure 5. C99 Selection

## Naming Conventions

The final naming convention shall comply with the following format:

Local and global variables:The tags for local and global variables shall contain the data type in lowercase at the beginning then, separated by an underscore is followed the variable name where the first letter of the first word shall be lowercase and the first letter of the second word shall be uppercase as showed in the following format:

datatype\_variableName

Functions:The tags for functions shall contain the description of the function, in other words, the name shall describe what the function is doing; where first letter of the first word shall be lowercase and the first letter of the second word shall be uppercase as showed in the following format

functionDescription

Code files**:** The code files shall be named in lowercase, in case of having two or more words they shall be separated by an underscore

file.h, file.c

All Documents: The final name of the documents is related to what it contains, the words are separated by an underscore and the first letter of each word shall be uppercase:

Software\_Development\_Plan.docx

# Testing

The purpose of testing is to incrementally integrate the software builds into larger software components, and components into a complete system. Testing is performed to validate each component’s ability to meet its stated requirements and to ensure the operability of the final software component.

Integration continues until all software components are integrated with the system-level hardware into a single functioning system.

## Verification strategy (BlackBox Test)

BlackBox tests will be performed on the Renesas target to verify normal and robustness system level requirements and high level requirements. Each test will consist on a procedure with steps to follow in order to perform the verification among with expected results to obtain, as well as a trace to the requirements tested.

Test results will be considered as ‘FAIL’ when the actual results of the performed test do not match the ones expected, and as ‘PASS’ when they do. The failure of a test can imply one of the following: the requirements are not correct, the code is not correct, the test needs to be modified in order to fully cover the requirement or an additional test is needed.

The BlackBox tests can be found at the following path *<PROJECT\_PATH>/4) Verification/BlackBoxTest*

## WhiteBox Strategy

WhiteBox test will be performed to ensure that the software components integrate correctly. The goal of this activity is to verify that the software interactions occur as expected by the software design and requirements and that the couples between components are behaving appropriately.

This activity will verify high or system level requirements which verification cannot be achieved through black box testing. As well as Black Box verification strategy, White Box tests will consist of a procedure of steps to follow and expected results. Test results will be considered as ‘FAIL’ when the actual results of the performed test do not match the ones expected, and as ‘PASS’ when they do. The failure of a test can imply one of the following: the requirements are not correct, the code is not correct, the test needs to be modified in order to fully cover the requirement or an additional test is needed.

The WhiteBox tests can be found at the following path *<PROJECT\_PATH>/4) Verification/WhiteBoxTest*

## Cyclomatic Complexity Redundance Index

# RELEASE

The Software Release Files are organized as defined below:

1. Development – Used for internal software releases.
2. Engineering – Used to be implemented, verified and validated in the hardware.
3. Production – Official release to the client

The weekly documents releases will follow the format of DocumentName\_Year\_Month\_Day

Development Release Files will be uploaded to the GIT *Dev* Branch each week with the following documents:

* C Code File .c , .h

Engineering Release Files will be uploaded to GIT *Dev* Branch each week with the following documents:

* Executable File .exe

The Production Release Files will be uploaded to GIT on the *Master* branch on October 26, 2019 containing all the documents and deliverable builds content defined in Deliverables Section of this document.

## Integration Test Strategy

The Integration Strategy for this project consist in an ascendant integration where each module is integrated once it is developed and tested individually. It starts when the ADC and the Display configurations are done. Then the sensor and motor are integrated and at the end the controlled is incorporated to the system.

Once the hole system is integrated, it shall run during an hour to validate the integration.

The Integration Results can be found at the following path *<PROJECT\_PATH>/* *4) Verification/Results/IntegrationTesting.docx*

## Validation Testing / Functional Testing

The BlackBox and WhiteBox tests cover this section. Refer to section Testing.

## Throughput and Flash and RAM measurement

Flash memory usage is calculated by RENESAS while building a project and is displayed in the Console. This value can be obtained by adding the *text* and *data* values.

Thus, the final Flash memory usage for this project was

Similar to the Flash memory, RAM usage is calculated by RENESAS while building a project and is displayed in the Console. In this case, the usage is obtained by adding the *bss* and *data* values.

# Results

# Lessons Learned