##### Project Charter

This project charter was created for firmware over the air Graduation project.

# Project Charter

|  |  |  |
| --- | --- | --- |
| 1.0 Project Identification | | |
| **Name** | *F.O.T.A (Firmware over the air).* | |
| **Brief Description** | *Firmware Over-The-Air (FOTA) is a Software Management (MSM) technology in which the operating firmware of a device is wirelessly upgraded and updated by its manufacturer. The process usually takes several minutes, depending on connection speed and file size.* | |
| **Supervisor** | Eng. Ahmed Al-Ashmawy - Eng. Ahmed Tourkey | |
| **Project Team members** | 1. Ahmed Adel Andeel. 2. Amr Ibrahim El-Nobe. 3. Marcelle Samir Isaac. | 1. Marina Medhat Zekry. 2. Mohanad Fawzy Sallam (leader). 3. Mostafa Nader El-Tourkey. |

|  |
| --- |
| 2.0 Business reasons for project |
| * Allows manufacturers to repair bugs in new units easily. * Allows manufacturers to remotely install new software updates, features and services - even after a device has been purchased. * Reduce costs over the manufacturers and provide a fast solution for generating a diagnostic reports. |

|  |
| --- |
| **3.0 Project OBJECTIVES (purpose)** |
| * Overall, Implementing the F.O.T.A concept. * Create prototype for the F.O.T.A concept using PC application, cloud server, Raspberry-Pi and microcontroller(s). * Enhance the security level of the cloud server. * Create a software communication protocol while flashing between raspberry-pi and the target microcontroller. * Create interactive PC-application with the user and report the flashing process status. * Understand the bootloader concept and creating applications suitable for on application concept. |

| **4.0 Project Scope** |
| --- |
| * Embedded Interfacing. * Software communication protocols. * Python utilities. * Embedded Linux. * Microprocessor Architecture and memory management. |

|  |  |
| --- | --- |
| **5.0 key Project DELIVERABLES** | |
| PC Application | PC Application that uploads .elf file, selects the desires target and displays |
| Raspberry pi system | Raspberry-pi fetches .elf file from cloud server, parses it and flashes it to microcontroller. |
| Bootloader | The bootloader flashes the application on the micro-controller |
| Project proposal | Project initial vision before the implementation phase. |
| Project charter | Formal documentation for The scope, objectives, and people involved in a project. |
| Guidelines | Steps of the project implementation. |
| Tools and resources | List of tools used for the project |
| Doxygen Reports | Reports generated automatically by Doxygen. |
| Cloud Server | Google cloud server for communication between Raspberry-pi and the PC application. |

| **6.0 Milestone dates** | | |
| --- | --- | --- |
| **Item** | **Major Events / Milestones** | **Dates** |
| 1. | Project initial version. | 30/05/2020 |
| 2. | Update communication protocol between Raspbery-pi and bootloader. | 06/06/2020 |
| 3. | Update Raspberry-pi to work automatically with several mc’s. | 06/06/2020 |
| 5. | Adding extra features for the system (flash new configuration - Diagnostics - flash from mobile application). | 13/06/2020 |
| 6. | Project final review and testing. | 19/06/2020 |

| **7.0 Project’s criteria for success (must be measurable)** |
| --- |
| * Flash an application from PC application to micro-controller. * Flash application to more than one type of micro-controller. * Ensure security connection between Raspberry-pi and cloud server. |