IOT Based Microgrid Automation

R.Vanitha[1], P. M. Kalpana[2]
[1,2]*Assistant Professor, Department of Electrical and Electronics Engineering*
R.M.D. Engineering College, Kavaraipettai

Abstract- Microgrids consist of low voltage distribution systems with distributed energy resources (DER) and controllable loads. The aim of the paper is to operate connected to the medium voltage grid or islanded in a controlled and coordinated way. Microgrids are controlled units that report their current state to a central control unit called Central Protection Centre (CPC) thus increasing the reliability of the system. Microgrids feature will increase the controllability, security and ease of power flow. By connecting load centers to a microgrid, uninterrupted power is supplied to households and industries with a reduction in energy theft, interruptions and loss in energy. However, the challenge lies in automating the entire process of monitoring, protection and control of all the grid parameters. This becomes more critical with the inclusion of DERs as they are inconsistent sources of energy. In this paper the concepts of the Internet of Things(IoT) is used to solve the issues of microgrid reconfiguration occurring due to faults, changing energy usage patterns and inclusion and removal of DERs. By using suitable processor based technology it is aimed to automate the process, reduce the cost and size of establishment and reduce the overall cost of energy to the supplier.
Keywords – Distributed energy resources (DER), Central Protection Centre (CPC)

I. INTRODUCTION

A microgrid is a small-scale power grid that can be operate group of interconnection of loads independently or collaboratively with other small power grids. Any small-scale, localized power plant that has its own generation and storage resources and definable boundaries can be considered a microgrid. A microgrid is distributed to the energy resources defined by electrical boundaries that act as a single controllable entity with respect to the grid.

Microgrids would involve a combination of resources, sometimes a quite complex one. There is no specific control on the size of microgrids. Instead, microgrid definitions focus primarily on two features:

It is a locally controlled system and it can be function both connected to the traditional grid (megagrid) or as an electrical island.

Microgrids are used in remote areas to power off-grid villages, military operations or industrial projects. They're being used in cities or towns, in urban centers, on university or corporate campuses, in hospitals or at data centers. Advantages to customers and utilities, i.e. increasing energy efficiency, minimize of overall energy consumption, environmental impact, improved the reliability of power supply, network operational benefits such as loss reduction, congestion relief, voltage control, or security of supply and more cost efficient electricity.

Distributed energy resources in microgrids used to produce electricity and load control technologies that can be typically located at customer sites and operated for the customer's benefit. Distributed Energy Resources can be included micro turbines, fuel cells, photovoltaic systems, and traditional internal combustion engines. It can be small-scale power generation sources located close to where electricity is used to provide an alternative electric power grid.

II. BASIC COMPONENTS OF MICROGRID

Figure.1 Structure of Microgrid

2.1 Local Generation:
A microgrid presents different types of generating sources that could be feed electricity, heat, and cooling to user. These sources are divided into two major groups – thermal energy sources and renewable generation sources (e.g. wind turbines, solar).

2.2 Consumption:
Microgrids are refer to elements that can be consumed electricity, heat, and cooling which the range from single devices to lighting, heating system of houses, and commercial centers. In this case of controlled loads, the electricity consumption could be modifying in demand of network.

2.3 Energy Storage:
In microgrid and energy storage are perform multiple functions, such as ensuring power quality, including frequency and voltage regulation, smoothing the output of renewable energy sources, providing backup power for the system and cost optimization. It included all electrical, pressure, gravitational, and heat storage technologies. When different types of energy storages with various capacities are available in a microgrid, it is preferred to coordinate their charging and discharging such that smaller energy storage does not discharge faster than those with larger capacities. This can be achieved under a coordinated control of energy storages based on their state of charge. A possibly working on different technologies energy storage system is used and they are controlled by a unique supervising unit called Energy Management System - EMS.

2.4 Point Of Common Coupling (PCC)
It is the point in the electric circuit where a microgrid is connected to a main grid. Microgrids that do not have a PCC are called isolated microgrids which are usually presented in the case of remote sites (e.g., remote communities or remote industrial sites) where an interconnection with the main grid is not feasible due to either technical or economic constraints

III. BLOCK DIAGRAM AND CIRCUIT DIAGRAM OF MICROGRID

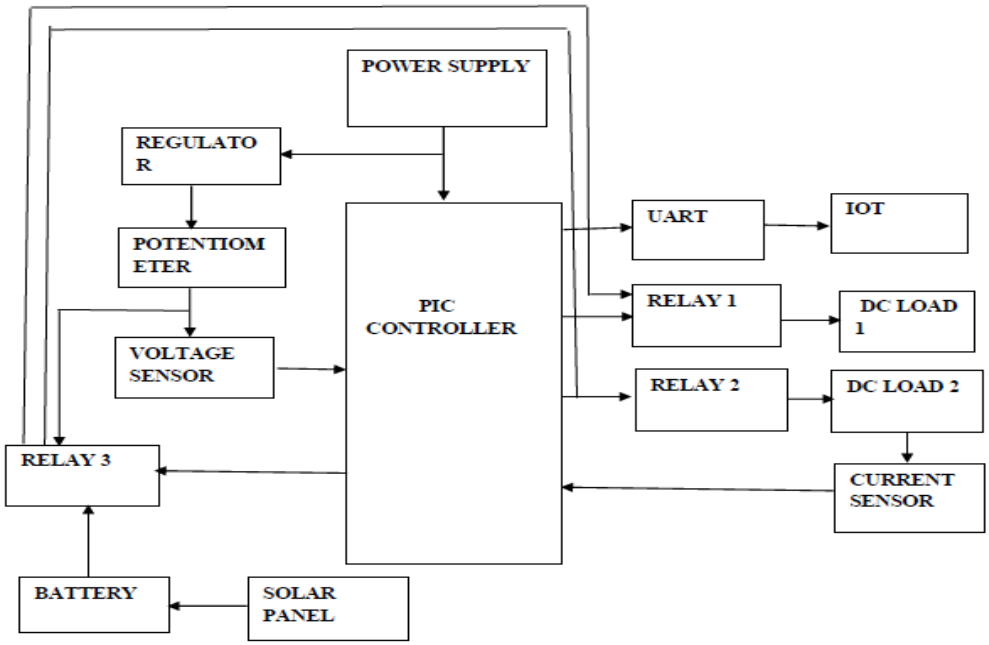

Figure. 2 Block diagram

In figure 2 shows the block diagram of microgrid automation. A relay consists of one sensing unit and one shifting unit is utilized. The sensor unit is sensing the following condition such as over current, over voltage, and under voltage. The data from the sensors are sent to the microcontroller unit for analysis and processing. The values are obtained and compared with the threshold values fed on to the database, so that any abnormal behavior of the system

is notified. An immediate action is followed by the system to rectify any abnormality detected. The detection and deletion of loads happens synchronously, which maintains the continuity and uninterrupted power supply.
The extended reach of IoT helps in overcoming such complications and also monitoring of the system with energy consumption becomes effortless. The IoT module detects the location, number of loads operating, amount of energy consumed from the supply and microgrid and the sensor values.

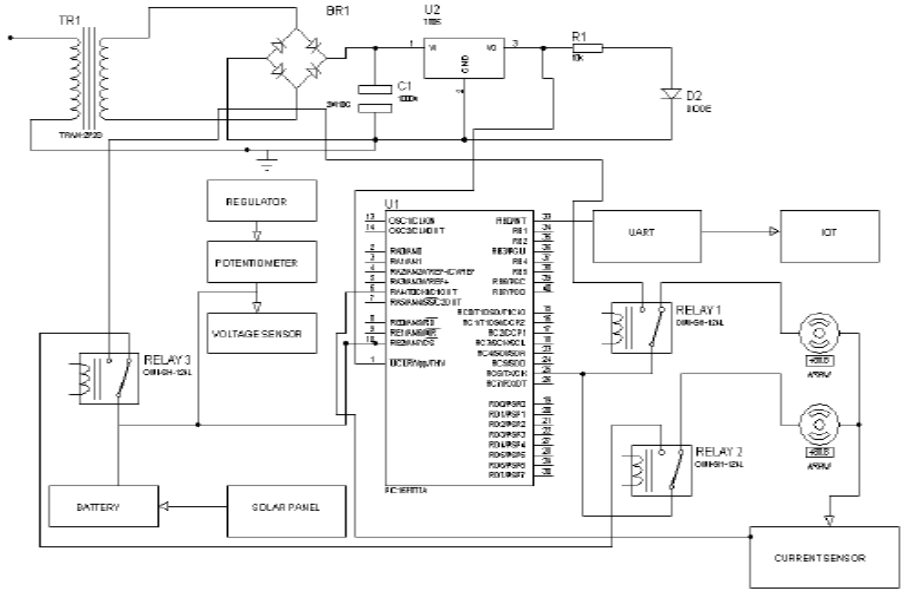

Fig.3 Circuit Diagram

IV. SIMULATION WORK

4.1 Description:
Embedded C is designed to bridge the performance between Standard C and the embedded hardware and application architecture. It extended the C language that are needed by signal-processing applications and that are commonly provided by DSP processors. The design of the fixed-point data types and named address spaces in Embedded C is based on DSP-C. The development of DSP-C by ACE, cooperation was sought with embedded-application designers and DSP manufacturers.

4.2 Named Registers:
Embedded C allowed direct access to processor registers that are not addressable in any of the machine's address spaces. The processor registers defined by the compiler-specific, named-register, storage class for each supported processor. The processor registers that are declared by conventional C variables. Developers using Embedded C could develop their applications, including direct access to the condition code register and other processor-specific status flags, in a high-level language, instead of inline assembly code. The responsibilities of computing data types array and structure offsets and all those things that C compilers routinely and easily do from developers to compilers.

4.3 I/O Hardware Addressing:
Embedded C is to improve the portability of device- Embedded C improves the portability of the device-driver code. The principle of the hardware device driver should only be concerned with the device itself. The driver operated on the device through device registers, which are device-specific. The method is to access the registers that can be very different on different systems, even though it is the same device that is connected. The I/O hardware is accessed primitives aim to create a layer that abstracts the system-specific access method from the device that is accessed. The goal is to allow the source-code portability of device drivers between different systems. In the design of the I/O hardware-addressing interface, three requirements needed to be fulfilled:
1. The device-drive source code must be portable.
2. The interface must not prevent implementations from producing machine code that is as efficient as other methods.

3. The design should permit encapsulation of the system-dependent access method.
The design is based on a small collection of functions that are specified in the <iohw.h> include file. These interfaces are divided into two groups; access to the device, and maintains the access method abstraction itself.
To access the device, the following functions are defined by Embedded C:

```
unsigned int iord( ioreg_designator );
void iowr( ioreg_designator, unsigned int value );
void ioor( ioreg_designator, unsigned int value );
void ioand( ioreg_designator, unsigned int value );
void ioxor( ioreg_designator, unsigned int value );
```

These interfaces are provided read/write access to device registers and typical methods for setting/resetting individual bits. Variants of these functions are defined to access the array of registers. Variants are also defined to operate with long values. All of these interfaces taken an I/O register designator ioreg_designator as one of the arguments. The register designators are the abstraction of the real registers provided by the system implementation and hide the access method from the driver source code. Three functions are defined by the managing I/O register designators. The abstract is entities for the device driver, the driver does have the obligation to initialize and release the access methods. These functions did not access or initialize the device itself because that is the task of the driver.

```
void iogroup_acquire( iogrp_designator );
void iogroup_release( iogrp_designator );
void iogroup_map( iogrp_designator, iogrp_designator );
```

The iogrp_designator specified a logical group of I/O register designators. The typically would be all the registers of one device. The I/O group designator is an identified or macro that can be provided by the system implementation. The map variant allowed cloning of an access method when one device driver is to be used to access multiple identical devices.

4.4 Embedded C Portability:
Embedded C implies that the portability of Embedded C programs is not always guaranteed. Embedded C provided access to the performance features of DSPs, not all processors are equal and not all Embedded C implementations can be equal. An application required 24-bit fixed-point arithmetic and an Embedded C implementation provided only 16 bits because that is the native size of the processor. When the algorithm is expressed in Embedded C, it would not produce outputs of the right precision.
Embedded C do not overcome such discrepancies. Embedded C provided a great improvement in the portability and software engineering of embedded applications. The differences between performance-specific processors, there is a remarkable similarity in the special-purpose features that they provided to speed up applications. To writing C code with the low-level processor-specific support may at first appear to have many of the portability problems usually associated with assembly code. The limited experience with porting applications that use Embedded C extensions, an automotive engine controller application, a 24-bit special-purpose processor, to a general-purpose 8-bit free. The porting process was much easier than expected. The exercise was to identify the porting issues and it is clear that the performance of the special-purpose processor is significantly higher than the general-purpose target identified the porting issues. The performance of the special-purpose processor is significantly higher than the general-purpose target.

PIC CONTROLLER INTERFACING WITH UART

```
void trans(unsigned char ll);
void trans_str(const char *po);
void trans(unsigned char ll){
TXREG=ll;
while(TXIF==0);
TXIF=0;}
void trans_str(const char *po){
```

```
while((*po)){
trans(*po++);
}}
```

4.5 MPLAB IDE

Free integrated development environment (IDE) from Microchip to implement code for PICs
Latest version 8.60(recommended), In Lab 7.6
IDE and documentation (user guide) can be downloaded from the Microchip website
To open MPLAB IDE

Start → All Programs → Microchip → MPLAB IDE v8.60 → MPLAB IDE

4.6 Project Creation

A project must be created for implementation ◦ Specify your device
Create and edit your files
Compile and link your project
Program the device

To create a project

Project → Project Wizard…

Begin project and specify device
Select Microchip MPASM Tool suite
Create New Project File
Add project files
Finish project creation and return to IDE

Updating Source Code
Modify .asm files
Under 'Source Files' folder in project window
Open and edit files to implement new functionality
Additional files can be created and added

Building Projects :

- Project → Build All (or Ctrl+F10)

Output window indicates success or failure
HEX file generated when project is built

Programming the Pic:
Directly Downloading Hex files to the PIC Memory
Used to transfer HEX file to the PIC and begin program execution
Two methods (HEX file must be generated)
MPLAB IDE

Programmer → Select Programmer → PICKit3

V. HARDWARE RESULT

As the first step, a sensor is a hardware device that senses a signal to a change in a physical condition such as over current, over voltage, and under voltage. The data from the sensors are sent to the microcontroller unit (PIC) for analysis and processing the sensor signals. The output of the sensor analogous voltage is amplified by means of the operational amplifier. The values are obtained and compare with the threshold values fed on to the database, so that any abnormal behavior of the system is notified. An immediate action is followed by the system to rectify any abnormality detected. The detection and deletion of loads happens synchronously, which maintains the continuity and uninterrupted power supply. The extended reach of IoT helps in overcoming such complications and also

monitoring of the system with energy consumption becomes effortless. The IoT module detects the location, number of loads operating, amount of energy consumed from the supply and microgrid and the sensor values. The hardware setup displayed in figure 4.

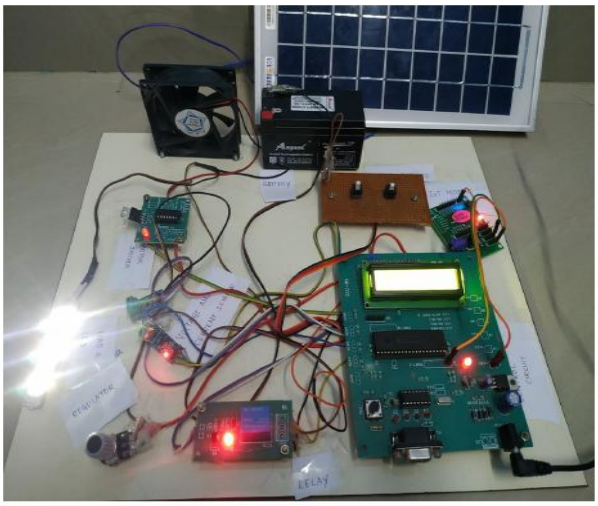

Fig.4 Hardware output

VI. SIMULATION RESULTS

The simulation results are shown in figure 5 and figure 6 , PIC microcontroller interfaced with UART, interfaced with LCD, to control the operation of PIC controller using MP Lab IDE.

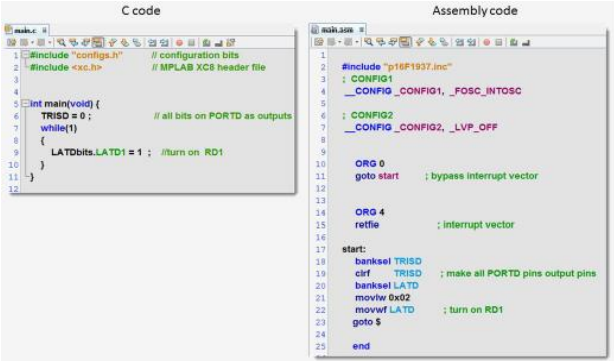

Fig.5 Interfacing with UART

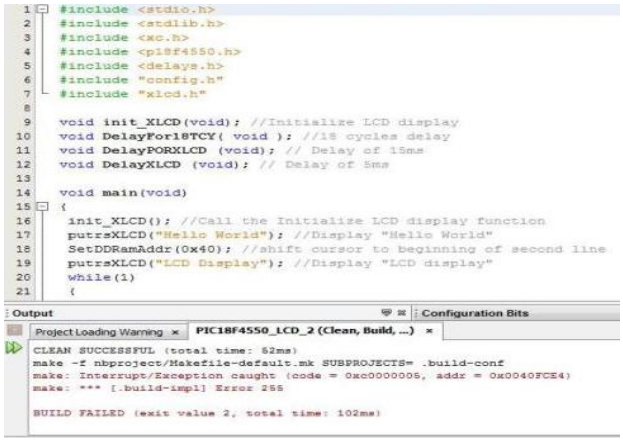

Fig.6 Interfacing with lcd

VII. CONCLUSION:

Thus the system successfully develop the continuous monitoring the automation of microgrid using IoT. An advanced system was developed for the Microgrid automation which overcomes the stigma of the Existing technologies of DC line microgrids. Due to its hopeful results, it can be ensured that the system come about can be applied on a large scale. It does not require any new expensive products as the proposed method can be utilized for system set up. This system can provide continuous DC supply to the load without any interruption and power loss . It is cost efficient and simple way of implementing the system. This technology can provide real time monitoring of power consumed by the loads. Fault in the microgrid and supply can be determined rapidly. The extended reach of IoT helps in overcoming such complications and also monitoring of the system with energy consumption becomes effortless.

VIII. REFERENCE

[1] Y. M. Lu, D. Liu and Y. H. Huang, "Feeder modeling and application based on CIM," Proceedings of the CSEE, pp. 157-163, 2012.
[2] J. M. Zhang, H. Chen and M. L. Zhou, "Generation of uniformly distributed distribution network connection diagram and perception measurement for situation awareness diagram," Automation of Electrical Power Systems, vol. 38, no. 13, pp. 166-173, 2014.
[3] B. X. Zhou, Z. Y. Meng and W. X. Wu, "Power grid automatic maping algorithm for radial distribution feeder," Electric Power, vol. 48, no. 12, pp. 136-140, 2015.
[4] J. Liu, Y. Wu and G. Q. Liu, "Conversion form GIS based feeder maps to electric diagrams," Automation of Electrical Power Systems, vol. 29,no. 14, pp. 73-77, 2005.
[5] I. Lendak, A. Erdeljan, D. Čapko and S. Vukmirovic, "Algorithms in electric power system one-line diagram creation," Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on, Istanbul, pp. 2867-2873, 2010.