Design and implementation of operating system for modular microgrid

Zhang, Longqi

2017

Zhang, L. (2017). Design and implementation of operating system for modular microgrid.
Master's thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/70637>

<https://doi.org/10.32657/10356/70637>

NANYANG
TECHNOLOGICAL
UNIVERSITY

Design and Implementation of Operating System for Modular Microgrid

by

ZHANG LONGQI

A Thesis

Submitted to the Faculty
of the

NANYANG TECHNOLOGICAL UNIVERSITY

in partial fulfillment of the requirements for the degree of
Master of Engineering

in

School of Computer Science and Engineering

2016

ABSTRACT

Power Park comprised of multiple modular microgrids exhibits advantages including better control flexibility, higher scalability, higher reliability, etc. The Energy Management System (EMS) of the Power Park can be divided into three layers, namely, converter layer, microgrid module layer and power park layer. We designed a two-layer communication network which connects the three control layers and corresponding EMS function blocks like Economic Dispatch (ED) and Unit Commitment (UC). Conventional EMS software can be only deployed on Windows PC with specified costly accessories and licenses which blocked the miniaturization and economization of the modular microgrid. In order to realize system control objectives in different layers, we designed and implemented a cross-platform agent-based microgrid Operating System (OS). Its components including Agent, Active Application Program Interface (API), Passive API and Graph User Interface (GUI) are designed and implemented to coordinate various converters based on the communication network. We implemented the real-time controller as the main component of the microgrid OS for executing functions like secondary voltage/frequency regulation, contingency analysis, etc. to enhance control accuracy within the microgrid.

TABLE OF CONTENTS

ABSTRACT	ii
LIST OF FIGURES	v
LIST OF ABBREVIATIONS	vii
CHAPTER	
1. Introduction	1
1.1 Background and Motivation	1
1.2 Contribution	2
1.3 Organization of the thesis	3
2. Background and Literature Review	4
2.1 Hybrid AC/DC Microgrid	4
2.2 Modular Microgrid based Power Park	6
2.3 Energy Management System	11
2.4 Communication	11
2.4.1 Industrial Communication Protocol	12
2.4.2 IEC 61850	14
2.4.3 RESTful API	15
3. System Design	16
3.1 Control Strategy	16
3.2 System Architecture	21
3.3 Design Philosophy	23
3.4 Data Flow	25
3.5 Minimum system	27
3.5.1 Agent	27
3.5.2 Real-time Controller	28
4. GUI and API	35

4.1	GUI	35
4.2	API Design	38
4.2.1	Active API	39
4.2.2	Passive API	44
5.	Communication Network	52
5.1	Overview	52
5.2	Field Area Network	52
5.2.1	CAN bus and OSI model	53
5.2.2	Application Layer Design	57
5.3	Park Area Network	65
6.	Conclusion and Future Work	67
6.1	Conclusion	67
6.2	Future Work	68
APPENDICES		68
A.	Author's Publication	69
A.1	Conference Papers	69
B.	Logic Node of Bidirectional Converter	70
BIBLIOGRAPHY		70

LIST OF FIGURES

Figure

2.1	Typical Microgrid Configuration	5
2.2	Multiple Hybrid Microgrid Modules Interconnection	7
2.3	Schematic Layout of Modular Microgrid based Power Park	8
2.4	Overall System Architecture Design of A Single Hybrid Microgrid Module .	10
2.5	Parallel VS Serial Communication	13
2.6	IEC61850 Layer Mapping [1]	14
2.7	RESTful API Design Example	15
3.1	Schematic Diagram for Hierarchical Control	17
3.2	Secondary Control Diagram of Microgrid Module Controller	18
3.3	Function Blocks for Tertiary Control	20
3.4	Function Architecture	22
3.5	Microgrid OS Components Relation	24
3.6	Data Flow	26
3.7	Agent	28
3.8	Converter Agent API Design	29
3.9	Real-Time Controller	30
3.10	Timed Automata of Traffic Light	32
3.11	Timed Automata of Real-Time Controller	33
4.1	Home Page for Microgrid Module	36
4.2	Page for Bidirectional Converter	36
4.3	Page for Battery Converter	37
4.4	Page for Photovoltaic (PV) Converter	37
4.5	Active Communication	40
4.6	Passive Communication	41
4.7	JavaScript Object Notation (JSON) Data File for Bidirectional Converter . . .	43
4.8	JSON Command File for Bidirectional Converter	44
4.9	Representational State Transfer (REST)ful API	45
4.10	Reading Status of Agent by RESTful API	46
4.11	Controlling Agent Without Parameters by RESTful API	46
4.12	Controlling Agent With Parameters by RESTful API	46
4.13	Logical Devices and Logical Nodes for DER Overview [2]	48
4.14	Basic Rules	48
4.15	LN for Bidirectional Converter	49
5.1	Communication Network	53

5.2	Open Systems Interconnection (OSI) Data Flow	54
5.3	Encapsulation	55
5.4	CAN Bus in OSI Model	56
5.5	Functions in CAN Bus Layers	57
5.6	CAN Bus 2.0A Frame	58
5.7	CAN ISO11898-2 High-Speed Network [3]	58
5.8	CAN ISO11898-3 Low-Speed Fault Tolerant Network [4]	59
5.9	CAN Bus Connection Microgrid Module	60
5.10	CAN Bus Frame Protocol	62
5.11	Binary Data in CAN bus	63
5.12	Data Transmission in CAN bus	64
5.13	Waveform of CAN Message Conversation	64
5.14	Park-Area Network (PAN) of the communication network	66

LIST OF ABBREVIATIONS

AC	Alternating Current
AGC	Automatic Generation Control
AJAX	Asynchronous JavaScript and XML
ANN	Artificial Neural Network
AOP	Agent-Oriented Programming
API	Application Program Interface
BBB	Beaglebone Black
CA	Contingency Analysis
CAN	Controller Area Network
DC	Direct Current
DERs	Distributed Energy Resources
DG	Diesel Generator
DLC	Data Length Code
DSP	Digital Signal Processor
DSPs	Digital Signal Processors
ED	Economic Dispatch
ED	Economic Dispatch
ELM	Extreme Learning Machine
EMI	Electro-Magnetic Interference
EMS	Energy Management System
ESS	Energy Storage System

FAN	Field-Area Network
GPIO	General Purpose Input/Output
GUI	Graph User Interface
HAL	Hardware Abstraction Layer
HMI	Human Machine Interface
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ID	Identifier
IED	Intelligent Electronic Device
IoT	Internet of Things
JSON	JavaScript Object Notation
LED	Light Emitted Diodes
LIN	Local Interconnect Network
LN	Logical Node
MPPT	Maximum Power Point Tracking
OOP	Object-Oriented Programming
OPF	Optimal Power Flow
OPF	Optimal Power Flow
OS	Operating System
OSI	Open Systems Interconnection
PAN	Park-Area Network
PV	Photovoltaic
RES	Renewable Energy Source
RESs	Renewable Energy Sources
REST	Representational State Transfer
RTC	Real-time Controller
SCADA	Supervisory Control and Data Acquisition
SE	State Estimation

SERTD	Sustainable Energy Research & Technology Demonstrator
SMES	Smart Multi-Energy System
SSL	Secure Sockets Layer
UC	Unit Commitment
UEMS	Universal Energy Management System
URI	Uniform Resource Identifier
USB	Universal Serial Bus
VPN	Virtual Private Network
WAN	Wide Area Network
WT	Wind Turbine
WTGs	Wind Turbine Generators

Chapter 1

Introduction

1.1 Background and Motivation

A microgrid is defined as a self-sufficient localized energy system of distributed energy resources, energy storages and electricity users that may or may not be interacting with the main grid. It is a promising technology that can improve the reliability and economics of energy supply to low-voltage distribution networks and end consumers. Nowadays microgrid development is shifting from prototype demonstration and pilot projects to full-scale commercial deployment. Based on the nature of the renewable energy source, microgrids are also widely implemented for integration of power electronic converter based renewable energy sources. Microgrids can be used for applications in remote villages, islands, offshore platforms, aircraft, etc. Microgrids can be controlled to operate in the grid-tied mode. In case of utility grid fault/failure, island mode operating can also be realized to enhance system reliability. Conventional microgrid solutions are usually customized based on the actual deployment environment. Professional personnel are always needed for operation, maintenance and modification. As a result, microgrids are usually too costly for large-scale and wide applications. Modularization of microgrid is introduced in this thesis to simplify deployment, enhance scalability and reduce cost.

EMS is a critical component that can help microgrid come to fruition. It optimally allocates the power output among all kinds of energy source units, economically serves the

load, responses to the operating transition between interconnected and island modes and automatically enables system synchronization between several microgrids. In general, a sophisticated microgrid energy management system has to operate and coordinate a variety of sources and loads in order to provide reliable, high-quality and efficient power to electricity consumers.

There are many EMS manufacturers such as *etap EMS*, *PowerOn Reliance*, *Schneider* and so on. But none of them can meet the demands of modular microgrids which can be interconnected to form a power park which is a family of microgrid system interlinked to share resources to take advantage of the availabilities of different forms of renewable energy at different locations and times.

1.2 Contribution

The main contribution is that we designed and implemented a cross-platform agent-based microgrid OS for modular microgrid. The OS is divided into three components:

- Agent. This is an abstract encapsulation of converter or any field device that needs to exchange data with the central controller by communication channels.
- Real-time controller. It is the most important component that provides most basic services like timing sequence control, task management, Contingency Analysis (CA), API for ED and UC, and so on.
- Real-time web-based GUI. It is implemented for remote monitoring and control.

Another contribution is that we modeled the most important converter which is the bidirectional converter that controls the energy flow and balances the power between Alternating Current (AC) and Direct Current (DC) buses in the microgrid module. The model is used for communication and API.

The last contribution is that we designed and implemented a two-layer communication network which provides high-performance and reliable communication between different field

devices in a microgrid module and plays a significant role in the interconnection between multiple microgrid modules.

1.3 Organization of the thesis

This thesis provides a background to the design of the hybrid microgrid module based power park and current situation of EMS in Chapter 2. This includes the hardware configuration of hybrid AC/DC microgrid and power park and the idea of how microgrid modules can be interconnected to a power park. A briefing of EMS and communication protocols is also included in Chapter 2.

The next three chapters cover the work done in the microgrid OS design including *agent*, *real-time controller*, *real-time web-based GUI* and *two-layer communication network*.

Chapter 3 introduces the system architecture which includes system design, design philosophy and data flow between the components in the system. Minimum system including *agent* and *real-time controller* is also introduced in this chapter.

Chapter 4 introduces the GUI module design and two types of communication API. GUI is designed as an independent module with a two-way communication with minimum system. API is the most significant component that makes the building block design and also the key to implementing interconnection of microgrid modules.

Chapter 5 introduces the two-layer communication network that supports API and connects the different components in the system.

This thesis concludes in Chapter 6 by highlighting the contributions of this work as well as some advice for future improvements in this area.

Chapter 2

Background and Literature Review

2.1 Hybrid AC/DC Microgrid

More and more renewable power sources, such as photovoltaics and wind turbines, are now connected in low-voltage distribution systems which operate as distributed generations or as AC microgrids due to resource depletion, energy crisis and ecological environment deterioration problems caused by coal or gas fired generators.

The distributed renewable energy sources are usually connected to local AC utility grid to supply local loads. Long-distance high-voltage transmission is not suitable in this scenario, so AC doesn't have significant advantages. In an AC microgrid as shown in Figure 2.1, DC power from PV and fuel cell systems is converted into AC using DC/AC inverters and Wind Turbine Generators (WTGs) are usually connected into the grid using a power conversion technique. In such case, DC loads are connected to AC microgrids using AC/DC converters. The DC microgrids have also been proposed to reduce energy loss during the conversion from DC to AC. However, DC/AC inverters are needed for AC sources and AC load in a DC grid. Therefore, the efficiency is reduced significantly due to multistage reverse conversions in an AC-only or a DC-only microgrid.

It is important to note that today's fast changing of electrical load toward electronic load which is DC in nature has led to the rapid development of DC grids. By 2015, Solar power had covered more than 1% of global electricity demand. In three countries in Europe (Italy,

Figure 2.1: Typical Microgrid Configuration

Germany and Greece) solar power supplied more than 7% of the electricity demand. China was the fastest growing market [5]. Light Emitted Diodes (LED) and Electrical Vehicles grow quickly in recent years. These mixed DC and AC renewable energy sources and loads are distributed in nature and are hence ideal platforms to introduce a hybrid AC and DC microgrid system which can alleviate the disadvantages of the current power grid structure.

A hybrid AC/DC microgrid is a limited energy system made up of DC and AC distributed energy sources, loads and energy storages [6]. It enhances power quality, stability, reliability, sustainability and economic benefits and it can run continuously in off-grid or on-grid mode. Based on the real-time prices on the energy market, the microgrid operator can take advantage of the economic opportunity to buy or to sell energy depending on the on-site generation capability in the grid-connected mode. In the occurrence of utility power outage, the point-of-common coupling breaker can automatically use its own distributed energy source to supply the critical loads at least within the microgrid.

2.2 Modular Microgrid based Power Park

General architecture allows microgrids to be easily scaled up and down through adding or removing generators, storages, and loads. Such changes can be performed in a parallel and modular manner to achieve required power production and consumption levels. Customization always costs much more than standardization. As functional units, microgrid modules can be easily interconnected into different sized Power Parks to support the expansion of the existing system in an economical and environmentally friendly way. Modularization of the microgrid is used to simplify deployment, enhance scalability and reduce cost.

Power Park system, a family of microgrids based on modular microgrid, can be easily configured for different applications based on cost, environment and reliability requirements. Each module in the Power Park is a complete microgrid with three-phase AC and DC buses to supply AC and DC load in a larger area as shown in Figure 2.2.

The schematic diagram of the modular microgrid based Power Park is shown in Figure 2.3. In each microgrid module, Renewable Energy Sources (RESs) like solar PV and wind turbine

Figure 2.2: Multiple Hybrid Microgrid Modules Interconnection

are integrated with respective power electronic converters which are controlled with Digital Signal Processors (DSPs). Renewable Energy Source (RES) converters are normally scheduled to operate in Maximum Power Point Tracking (MPPT) mode to maximize the utilization. In order to compensate the intermittency of RES generation due to variations of solar irradiation and wind speed, localized Energy Storage System (ESS) is usually installed. Battery converter controls the ESS output power through charge/discharge to maintain the system power balance in real time. The microgrid module can operate in either grid-tied or island mode based on the status of utility grid relay. A resource manager is defined as a cluster of all sensors monitoring the input/output status and all relays which control the connections of respective sockets. The microgrid is also capable of interconnection with the neighboring microgrid modules to form a Power Park.

Embedded controller Digital Signal Processor (DSP) is implemented for control of the converters and the resource manager. Module-wise optimization objectives include maximization of RESs generation, bus voltage/frequency regulation, optimized scheduling for power exchange with utility grid, etc. Therefore, a microgrid module controller together with the communication link between the controller and DSPs are needed. Microgrid module controller acquires system operating status through the communication link. After execution

Figure 2.3: Schematic Layout of Modular Microgrid based Power Park

of the optimization algorithms in the micro grid module controller, the operating references are assigned to converters and the resource manager accordingly.

Microgrid modules can be interconnected to form a Power Park. Microgrid modules can take the neighbor as the spinning reserve or backup energy source to enhance system reliability. However, the microgrid module controller only has the observability within the module. Optimized coordination can hardly be realized without a Power Park controller. Therefore, Power Park layer controller is introduced and its communication interlink with all module controllers is proposed as shown in Figure 2.3. Power Park controller is responsible for globally optimized power flow scheduling among microgrid modules.

The advantages of the proposed modular microgrid based Power Park include:

- Plug-and-play. The microgrid module is an energy hub for integration of various energy sources and storages. Automatic detection and integration are embedded in the microgrid module controller.
- Highly scalable. Adding/removing of energy sources in an individual microgrid module or a complete module can be realized with minimum or no systematic modification.
- Multi-layer control. Different layers of control algorithms are realized in different hardwares and only necessary information will be acquired by the respective controller. It helps to prevent communication congestion and enhance the speed of response.

The overall architecture design is shown in Figure 2.4. The main components can be divided into Local Central Controller Module, Resource Manager Module, 4 types of Power Cards, DC Bus Bar, AC Bus Bar and Communication Bus Bar. Power Cards represent the integrated power electronics modules (IPEMs) for each power converter with plug-and-play feature. Each Power Card mainly consists of the main power circuit, the local DSP controller, the auxiliary power supply, the sensor circuits, protection circuits, and a simple user interface (LED lights indications and an LCD screen display). The Beaglebone Black (BBB) has been adopted as the execution element for the Local Central Controller Module. BBB is a low-power open-source single-board computer produced by TI. The Resource Manager Module is proposed

to manage resources like PV, wind turbine and diesel generator, battery, loads including DC loads and 1-ph/3-ph AC loads, and interfaces with other power systems. The Resource Manager Module integrates the functionality of sources/loads/bus-bar/ambient information collecting, protection/fault isolation, load shedding, and system structure reconfiguration.

Figure 2.4: Overall System Architecture Design of A Single Hybrid Microgrid Module

2.3 Energy Management System

EMS are critical components that can help the microgrid come to fruition. In general, a sophisticated microgrid energy management system has to operate and coordinate a variety of sources and loads in order to provide reliable, high-quality and more efficient power to electricity consumers. A typical EMS mainly consists of real-time *Supervisory Control and Data Acquisition (SCADA)*, *Generation Dispatch and Control* and *Energy Scheduling and Accounting*. Many EMS manufacturers such as *etap EMS* [7], *PowerOn Reliance* [8] and *Schneider* [9] have their own EMS solution in the smart grid area. But almost all of the EMS manufacturers are using Microsoft-only technology, such as ActiveX. Microsoft has been staying in the monopoly position on the operating system market for a long period and is not very popular in a Linux dominated Internet of Things (IoT) era. One limitation is that they are closed source software and you can not review the code to make sure they don't have 'back door'. EMS is a high security level software since it controls all the electrical devices. And developers must have complete control of the middle ware. Besides that, most EMS software come with their own costly devices so that this greatly affects the cost of microgrid that only contains several converters and tens of relays and is used in a family or business building. So a high-availability EMS must be developed for the modular microgrid which can be further interconnected to a Power Park so autonomy, stability, scalability, and economics can be achieved.

2.4 Communication

Reliable communication provides basic support to the microgrid OS and their interconnection. And the complex electromagnetic environment can not be ignored in the communication network design. Two-layer communication network is designed based on control strategies which will be introduced in Chapter 5.

2.4.1 Industrial Communication Protocol

Communication can be classified as *Serial* and *Parallel*, such as RS232 and PCI respectively. In a serial communication system, data is transmitted bit by bit sequentially from one side to another side. It will be 8bits (or integral multiple of 8 or 16) that are on different channels simultaneously and synchronized to a clock as shown in Figure 2.5. As a result, there is a speedup in parallel transmission bit rate over serial transmission bit rate. However, this speedup is a trade-off versus cost because multiple channels and multiple General Purpose Input/Output (GPIO) cost more than serial communication, and as the parallel cables get longer, the synchronization timing between multiple channels becomes more sensitive to Electro-Magnetic Interference (EMI) and distance especially in the microgrid module.

To resolve the cost, speed, distance and EMI limitations, many other serial transmission standards have been developed such as Universal Serial Bus (USB), RS485 and Ethernet. Among all of them, RS485, Local Interconnect Network (LIN) bus and Controller Area Network (CAN) bus are quite popular in the micro controller and DSP which is used as a converter controller.

Compared with RS485 and LIN bus, CAN bus has two layers in the OSI model, namely a physical layer and a data link layer that is responsible for error control, flow control and media access control. All these tasks in the data link layer save a lot of time for the higher level application and make communication much more stable and efficient. All of these make CAN bus suitable for real-time application.

Figure 2.5: Parallel VS Serial Communication

2.4.2 IEC 61850

There are several standards that can be applied to a microgrid, such as OpenADR, and EMIX as well as the IEC 61850 standard. IEC 61850 as the most popular microgrid standard has been used by many electrical device manufacturers for several years. Compatibility with existing devices can be very helpful for further popularization.

Standing on the successful OSI stack, standard describes the use of the Manufacturing Message Specification (MMS) standard as an application-level protocol as shown in Figure 2.6.

Figure 2.6: IEC61850 Layer Mapping [1]

Although IEC 61850 is officially described as a communication structure for substation and feeder equipment, its most useful contribution is the definition of an object model for all substation objects, such as relay and converter. IEC 61850 provides a structural decomposition of the components in the communication network. This means that each component of a microgrid or even itself can be described as a IEC 61850 model [10]. This virtual power plant concept makes it very suitable for our modular microgrid based Power Park.

2.4.3 RESTful API

RESTful API [11] typically, but not always, provides an application program interface over Hypertext Transfer Protocol (HTTP) or Hypertext Transfer Protocol Secure (HTTPS) with the same HTTP verbs (GET, POST, PUT, DELETE, etc.) that web browsers use to operate web pages and to send data to remote servers. A typical RESTful API is shown in Figure 2.7. Resources in application using REST API are identified by Uniform Resource Identifier (URI).

Figure 2.7: RESTful API Design Example

The uniform interface, the best part of REST, is fundamental for the design of RESTful API. It simplifies the design and has a similar structure to IEC 61850, which enables resources to be designed and operated independently as shown in Figure 2.7. In the example, a HTTPS request is sent from clients to the RESTful API server to turn off battery 0.

Chapter 3

System Design

3.1 Control Strategy

Hierarchical control [12] based on different time scales as shown in Figure 3.1 is proposed for system operation which is also the major reason for the communication network design in Chapter 5. EMS is divided into three layers, namely primary control, secondary control and tertiary control. The primary control is realized in converter local controllers (DSPs). Converter controllers are able to follow the scheduled operating mode and operating references. All converters in a microgrid module coordinate to realize functionalities like voltage/frequency regulation, maximum power tracking of RESs, load following, etc. In case of operating constraint (output power/current, bus voltage, etc.) violation, software protection will be activated to prevent damages to the respective converter. Thanks to the computational capability of the DSPs, the faults can be detected and eliminated within milliseconds.

In the primary control, converters operate based on local information. In order to eliminate the control errors which are induced due to line impedance and control inaccuracy, secondary control is implemented. It is mainly responsible for two objectives: a) tuning of converter operating references to enhance power quality and control accuracy; b) solving critical issues like load shedding, generation curtailment, etc. The secondary control is implemented at the second level due to the transmission delay involved.

Tertiary control is proposed to realize systematic optimization economically. Forecasting of

Figure 3.1: Schematic Diagram for Hierarchical Control

load consumption, RES generations based on algorithms like Artificial Neural Network (ANN), Extreme Learning Machine (ELM), etc. are implemented to indicate the future situation. UC, ED and Optimal Power Flow (OPF) are executed to generate the long-term operating references for all sources. The tertiary control is usually executed every several minutes.

The primary control, secondary control and tertiary control are designed to be implemented in the converter controller (DSP), microgrid module central controller (BBB) and Power Park controller (BBB), respectively. The information flow diagram of the microgrid module controller is shown in Figure 3.2. The microgrid module controller acquires module operating status from the field device controller, which controls power electronic converters and the resource manager, through the communication channel. All field devices' information will be stored in the respective agent, which acts as the intermediate between field devices and the microgrid module controller. On the other hand, whenever operating commands are generated, the references will be assigned to the respective controller through the agents.

We divided the secondary control into two parts: CA and short-term scheduling. CA

Figure 3.2: Secondary Control Diagram of Microgrid Module Controller

is specified to deal with module-level emergency issues. Whenever critical issues occur, an interrupt will be triggered to notify the central controller. Contingency analysis is activated to analyze the critical situation and the respective response procedure (which has been pre-scheduled) will be executed. The solutions for the critical issues include: load shedding, generation curtailment, operating mode selection and transitions, etc. Interrupt based notification ensures that the module controller is able to response immediately (within several milliseconds).

If nothing critical has occurred, short-term scheduling will be implemented. Short-term scheduling is mainly used to compensate the dispatch error due to forecast inaccuracy, and eliminate power tracking error and voltage/frequency deviations. The local database serves as the back-end of the EMS function block in this case. DSP information is transferred to the agents in the central controller to update the operating status every 1 second. The agent will store the data into the database. The secondary controller acquires system operating status from the database and generates the modified operating references. Again the operating references will be stored in database and the respective agent is responsible for assigning the operating references to field device controllers.

In the tertiary control, long-term scheduling is implemented. The flow chart of all the EMS function blocks in the tertiary control is shown in Figure 3.3. State Estimation (SE) and OPF are implemented to analyze system operating status and optimize the power flow to minimize transmission line power losses. Forecasting based on the historical data, which is available in database, is implemented to indicate the future load consumption and RESs generation. The information is sent to UC and ED to generate the output scheduling of all converters with time step of minutes. The long-term scheduling output will be converted to output references of different energy sources in each microgrid module through the secondary control.

Figure 3.3: Function Blocks for Tertiary Control

3.2 System Architecture

The secondary control is the fundamental function that connects every field device in the microgrid module. A microgrid module is an independent individual that can plug PV, Wind Turbine (WT), Diesel Generator (DG) and battery as a energy hub to feed the loads that are connected on the AC bus and DC bus. CA, load shedding and demand response are necessary functions that make all the field devices a robust microgrid.

On the other hand, a single microgrid module must have the capability of interconnection with other peer microgrid modules or a utility grid. Information exchange is the key to a safe and stable energy transferring.

To fulfill the above requirements, we designed a four-layer microgrid OS and the function architecture is shown in Figure 3.4. The four layers are defined as *Interface*, *Application & Data*, *Agent* and *Field Devices*. The field devices include converters and a resource manager. *Agent* is the basis that supports the whole system. *Application & Data* consists of all the microgrid applications such as ED, UC and CA. *Interface* is divided to *GUI* and *API*. The former is developed for microgrid daily routines for end users and converter control for experts. The latter is mainly used for high level integration and further development.

Figure 3.4: Function Architecture

3.3 Design Philosophy

The main idea of the application is Agent-Oriented Programming (AOP), compared with Object-Oriented Programming (OOP) which uses object providing member functions with variable parameters, AOP has specified agents with interface and message capabilities. *Agent* can be subset of *Object* or higher abstraction of *Object*.

In a microgrid there are several types of field devices as listed below:

- Bidirectional interlinking converter controller DSP can control the direction of the power direction between AC bus and DC bus, making sure that the microgrid has the best economic benefit and provides high quality power.
- PV and WT converter controllers try their best to draw power to support the microgrid based on the microgrid configuration.
- Battery converter controller is the most critical component in the microgrid. It serves a lot of valuable functions, including balancing the fluctuations in PV and WT power flow, providing power peak shaving, establishing the DC bus and initiating the minimum system. Storage can play a multi-function role in the microgrid module.
- Resource Manager is designed as a DSP-based super meter to obtain the output information of solar PV, wind turbine, battery energy storage, diesel generator and power exchange with the utility grid and other modules.

Based on the type and functionalities of field devices, we designed field device agents that integrate with API that can be called remotely and execute some specific tasks based on its firmware and hardware configuration.

Besides the field device, there is one more special agent called Real-time Controller (RTC) that is designed as a representative of the microgrid so it can easily integrate with other microgrid modules.

RTC implements the basic module level functions including self-diagnosis, black

start/shutdown, real-time balancing, Automatic Generation Control (AGC) and RES Smart Integration.

GUI is designed as an independent module working with agents by API. JSON format is used between the agents and the GUI module to simplify the communication and make it much easier for debugging.

EMS function blocks also run as an independent module that interacts with the central controller by database only. The input of those function blocks is current microgrid configuration profile, current running status and historical data. Output of those function blocks is also put back to the database so RTC can retrieve it and use it as the reference of real-time balancing. The relation between these modules is shown in Figure 3.5.

Figure 3.5: Microgrid OS Components Relation

Agent is the bridge that connects the field devices with a higher level control so other modules in the system only need to interact with field devices by calling agent API instead of taking care of the communication details. Adding and removing field devices also become much easier.

The other modules like GUI get access to the field devices by calling the RTC API rather

than dealing with every agent and taking care of the procedure of every task such as black start, adding PV and so on.

Agent API also can be called directly in the debug mode and expert mode so the field device development and debugging can be more convenient.

Agent and RTC are the minimum autonomous system of the microgrid OS. Agent receives the field device status and transmits corresponding commands. RTC maintains the basic functions and deals with fault conditions.

The next section will illustrate the data flow between these components in detail.

3.4 Data Flow

Based on the functionalities of each module and time restriction for each function, three main modules, namely Core, EMS and GUI, are implemented in order to fulfill the system requirements. GUI and EMS take charge of GUI/Human Machine Interface (HMI) and long-term scheduling respectively. Core including all agents and RTC is the minimum system that implement the secondary control and control field devices. Detailed information flow among the three modules is shown in Figure 3.6 below.

Event-driven mechanism is applied in this system and there are three event sources:

- CAN bus Message. It is the base event source that comes from field devices and the high level decisions are made based on the device status and fault messages. CAN bus is part of the communication network which will be covered in Chapter 5.
- Timeout. This is mainly used to trigger the alarm system if the hardware or communication fails. A proper threshold is set for every task, sub-task and agent, so RTC can handle this and the operator can get alarmed in time when any hardware has no response.
- External Input (User, Universal Energy Management System (UEMS) or Peer Microgrid).

This category only accounts for a small fraction of all the events, but it always plays a

Figure 3.6: Data Flow

significant role in system controlling, such as microgrid on/off, transaction from island mode to grid-tied mode and electricity price changing.

To better utilize the communication channel, CAN bus messages from field devices are further divided to two categories as shown in Figure 3.6:

- *Periodic Message* includes all the peripheral statuses such as relay status and bus voltage. The update interval is fixed as 1 second because this is required by functions like ED and UC. A periodic message coming from a field device updates the corresponding agent model then the data will be sent to GUI and the database which is used for function blocks like ED and UC.
- *Fault Message* with higher priority than the periodic message includes over-voltage, over-current, etc. The system needs to respond as soon as possible based on the situation. Fault message directly goes to the RTC which will process the emergency situation, record it to the corresponding file and finally notify the GUI module.

The event determines the actions of the agent and RTC and forces the components to perform the corresponding tasks. The next section will explain how the agent and RTC work.

3.5 Minimum system

The minimum system including *agents* and *RTC* as listed below provides the most basic functions that a self-governed microgrid needs such as self-diagnostic and fault removal.

- Converter Agent. The abstract encapsulation of converters and the resource manager in the modular microgrid is shown in Figure 3.7.
- Resource Manager. This is a similar module to converter agent. The only difference is that the resource manager controls and monitors all the relays and sensors in the microgrid.
- RTC. It is a function block or a special agent for the whole microgrid module as shown in Figure 3.9, in which the detailed secondary control is implemented and also its API for the overall control.

3.5.1 Agent

Agents are abstract encapsulations of field devices and designed for simplifying the programming workload and the system. Every DSP controller has an agent mapping in the system. The data flow of agents is shown in Figure 3.7. Agent is further divided to *Agent Model* and *Agent API*.

Agent Model is a data structure representing the corresponding DSP status and gets updated by data coming from field devices at a fixed interval.

Agent API can be called by the GUI module as well as RTC by a communication channel. The GUI module and RTC control every field device by calling agent API rather than interacting with raw data.

Agent API provides public API services for RTC and GUI module as shown in Figure 3.8.

Figure 3.7: Agent

3.5.2 Real-time Controller

RTC conducts a secondary control which is divided to *contingency analysis* and *short-time scheduling*.

When critical situations are detected by the DSP controllers (converters and the resource manager), besides the immediate responses in the local controller for protection (stopping the converter, operating mode change, changing relay status, etc.), the critical situation will be sent to the central controller immediately to inform the status change. The detailed types of critical information have been documented separately.

Once the central controller detects the occurrence of any critical issue, contingency analysis will be activated. Detailed critical situation will be analyzed and corresponding function will be called to resolve the problem. Detailed critical situation and corresponding solution have been planned and programmed in the contingency analysis. The solutions in case of a critical situation include load shedding, generation curtailment, operating mode transition, etc. Details of the functions are documented separately.

Figure 3.8: Converter Agent API Design

If no critical issue occurs, the following functions of short-time scheduling are executed:

- **Real-time balancing:** Real-time balancing is implemented to reschedule the power output of all sources based on the actual load consumption and RES generation. It helps to compensate the forecasting error.
- **AGC:** AGC is mainly responsible for the secondary frequency and voltage regulation to minimize the voltage and frequency deviations.
- **RES smart integration:** RES smart integration is responsible for the automatic renewable energy source (PV and WT) integration and disconnection.

Figure 3.9: Real-Time Controller

Every function above is an individual task and independent of each other. There is a high possibility that multiple tasks are running at the same time. The response signals from field devices may come anytime, so we designed the microgrid operating system as an event driven system to make sure that the microgrid is highly efficient and real-time responsive. We designed the RTC as a Timed Automata [13] to manage the tasks above.

Timed automata is a finite automaton extended with a finite set of real-valued clocks. A timed automaton is a 7-tuple $TA = \langle L, L_0, C, A, G, E, INV \rangle$ in which:

- L is a finite set of locations.
- L_0 is a set of initial locations ($L_0 \subseteq L$).
- C is a finite set of clocks.
- A is a finite set of actions.
- G is a finite set of guards.
- E is a set of edges between locations ($E \subseteq L \times A \times G \times 2^C \times L$).
- $INV (L \rightarrow G)$ assigns invariants to locations with actions, guards, and a set of clocks to be reset.

Below is a timed automata for a traffic light. There are three locations (states) in the timed automata: "red" which is the initial location with double circles, "green" and "yellow". The location invariant is " $x < 5$ ", " $x < 5$ ", and " $x < 1$ " respectively. Green text on the edges (transitions) is the guard for the corresponding transition and light blue text is updated (actions to be done) during transition.

Figure 3.11 below shows the timed automata for a real-time controller.

Events coming from *CAN bus message*, *timeout* and *external input* will trigger the RTC and push it to the next state based on whether it's critical or not. RTC creates or updates corresponding tasks after receiving the critical event, and then checks task progress and the amount of time left. Tasks that are finished in time will finish the routine. Timeout and failed tasks will be destroyed and trigger the alarm system and the logging mechanism. Tasks that are still waiting and unfinished will finish the routine, update time and keep waiting for the next event.

The most basic function of a microgrid, like CA, operating scenario transition, etc. is built in the core (minimum system) with the agent and RTC. Agents represent the field devices in the operating system and RTC ties them together to balance power transferring, shed load and deal with all kinds of critical issues. But it's not enough if we want to make the microgrid module

Figure 3.10: Timed Automata of Traffic Light

Figure 3.11: Timed Automata of Real-Time Controller

high-usable and user-friendly. We designed API for microgrids interconnection and integration. GUI is also designed for operating the microgrid and tuning the converters. Both of them will be described in Chapter 4.

Chapter 4

GUI and API

4.1 GUI

GUI is the entry for normal operating and maintenance. It's also heavily used for field device debugging during the development period. To decouple with the core module (minimum system), we designed GUI as an independent module that interacts with the core module by API. Because of its web based character, GUI can be perfectly deployed in Windows, macOS, Android and any other device that supports HTML5.

The end users only need some basic functions like turn on, turn off and the microgrid developers need much more details about every field device, like DC bus voltage and PV power. Two different types of web page are designed for these two user groups.

The Main page of GUI shows the current microgrid configuration and some basic running status as shown in Figure 4.1. Most operations in the microgrid are simplified as on/off operation so the layman can easily operate the microgrid. The colors of icons including PV, battery and wind turbine indicate the devices' availability. The form and color of the switches indicate its on/off status. Power consumption is shown on the corresponding position.

GUIs of bidirectional converter, PV converter and battery converter are shown in the following Figure 4.2, 4.3 and 4.4. They have much more details. Converters can be fully controlled on these pages including isolating/connecting, on/off and other running parameters.

Figure 4.1: Home Page for Microgrid Module

Bidirectional Interlinking Converter ❤️

Figure 4.2: Page for Bidirectional Converter

Figure 4.3: Page for Battery Converter

Figure 4.4: Page for PV Converter

Agents and the real-time controller as the back-end of GUI provide real-time data to the corresponding converter page and execute the commands from the GUI module. The heart icon on the right side of the page title indicates the connection status between the GUI module and the core module. The connection content is the date and time of the last message from the field device when it loses communication. Converter isolation or connection function with power bus should be done by the converter itself automatically based on the running scenario. But we still designed an isolation/connection button on the right-top corner for all the converter pages so converter developers can manually control them during the tuning and debugging process. Because the control parameters of converters depends on different running modes, tabbed document interface is designed for control of the running mode. To get a better understanding of the converter running status, we designed a line graph to show power value. The line graph updates dynamically with the data pushed by the core module through active API which will be introduced in the next section.

With *GUI*, *Converter Agent* and *RTC*, a usable system that can provide basic functions of automation and protection is complete. To make the system interconnectable and cascable, API is a must-have component which will be introduced in the next section.

4.2 API Design

The microgrid module is designed not only to work independently but also to work with other microgrid modules or building management systems. Central controller is the one responsible for communication with the other microgrid modules which can be located far from each other. There are two categories for API:

- Active Communication. This is mainly used between the *GUI module* and the *core module* which pushes new data to GUI so the operator can get the newest microgrid status and alarm information in time as shown in Figure 4.5.

The GUI module requests for data communication connection after GUI on the client side is ready. Once the connection is established, the core module will keep sending newest

device statuses and alarm information to the GUI module. Commands from operators can be sent to the core module by the same connection without any disruption in several milliseconds.

- **Passive Communication.** Every microgrid in a Power Park needs to know the status and configuration of others so we designed the external communication method that is used between different microgrid modules or between any microgrid modules and other power systems as shown in Figure 4.6. Compared with the GUI module, external devices require a much lower transmission frequency so this API is designed at standby mode most of the time and give responses only after receiving a request.

4.2.1 Active API

Websocket as a low level protocol supports active API. Websocket is a protocol providing full-duplex communication channels over a single TCP connection. It is an advanced technology that makes it possible to start an interactive communication session between clients and a server. With websocket, the server can push data to clients continuously without pullings from clients. This feature takes less resource consumption than Asynchronous JavaScript and XML (AJAX) and conventional complete HTTP requests. Its another feature is native JavaScript support. Websocket depends on TCP which is the fourth layer of the OSI model and only the handshake phase is initialized by HTTP. To meet the requirement of microgrid information exchange, we designed a data presentation layer which is standardized to simplify the process and hide the details from the application layer. The figure below will show the data structure of the data presentation layer.

The data structure is shown in Figure 4.7. Considering the scalability of the communication network, one channel is set for one device category to reduce communication traffic because only one device is operated by GUI most of the time. There are four segments for each message pushed to GUI:

- *dev.* Device category is used to indicate the sender, (*bidirectional converter* in this case).

Figure 4.5: Active Communication

Figure 4.6: Passive Communication

- *number*. Number at line 3 is the sequence in the corresponding device category.
- *timestamp*. Timestamp is used for data validation and message sequence verification because the delay on the Ethernet may be relatively too long to miss the sequence with the next message and the timestamp is settled as soon as the message from Field-Area Network (FAN) is captured. This is also one of the important factors that ensures the plug-n-play concept.
- *data*. This is the actual data from devices. The name of them is predefined in the appendix.

Similarly the commands from the GUI module to the core module also use JSON format as shown in Figure 4.8. Compared to the data from the core module to the GUI module, the command message has two more segments:

- *cmd*. This is the API name defined in the core module. It decides which API will be called in the core module.
- *parameters*. This is one more segment which contains all the parameters that are defined by the core module API. The format still follows JSON format as a key-value pair.

```

1  {↵
2    .... "dev": "bic",↵
3    .... "number": 1,↵
4    .... "timestamp": 1470230260.547087,↵
5    .... "data": {↵
6        .... AC_Reference_frequency: 0.12↵
7        .... AC_Reference_voltage: 0.5↵
8        .... AC_VQ_droop_coefficient: 1.8↵
9        .... AC_bus_Frequency: 0.05↵
10       .... AC_bus_current_A: 0.24↵
11       .... AC_bus_current_B: 0.18↵
12       .... AC_bus_current_C: 0.14↵
13       .... AC_bus_voltage_phase_A: 1.6↵
14       .... AC_bus_voltage_phase_B: 0.5↵
15       .... AC_bus_voltage_phase_C: 0.5↵
16       .... AC_fP_droop_coefficient: 2.2↵
17       .... AC_output_active_power: 29↵
18       .... AC_output_reactive_power: 2↵
19       .... AC_relay_ON_OFF_status: 1↵
20       .... Active_power_reference: 4↵
21       .... DC_Reference_voltage: 1.4↵
22       .... DC_VP_droop_coefficient: 0.2↵
23       .... DC_bus_current: 0.5↵
24       .... DC_bus_voltage: 1.9↵

```

Figure 4.7: JSON Data File for Bidirectional Converter

```

1  {
2      "dev": 'bic',
3      "number": 1,
4      "cmd": "TURN_ON_RECTIFIER",
5      "parameters": {
6          "DC_Reference_voltage": 380.5,
7          "Reactive_power_reference_rec": 3.1
8      }
9  }

```

Figure 4.8: JSON Command File for Bidirectional Converter

4.2.2 Passive API

Passive API uses *tornado* as the back-end. Tornado is a Python web framework and an asynchronous networking library by using non-blocking network I/O. To make sure it is easy to use and to get better compatibility, we designed REST and IEC 61850 model for this microgrid OS.

4.2.2.1 REST API

To make a RESTful API, there are two things we need to do: One is to define the data structure to be exchanged. The other is the URI definition. The data structure is the same with the active API design which is shown in Figure 4.7, so we can reuse the code. We defined the URI based on the hardware configuration as shown in Figure 4.9. The complete URI consists of four parts:

- *Protocol*. The protocol is HTTP or HTTPS depending on the network environment and the security requirement.
- *IP or Domain Name*. This is the address of the service provider which is the central controller BBB of the microgrid module.
- *Port*. It is a "logical connection place" that is used to bind to the application process,

RESTful API in this case.

- *Resource ID*. ID identifies resource by the agent type and the agent number.

Figure 4.9: RESTful API

Based on our design, the RESTful API has three usage scenarios. Reading status from agents is shown in Figure 4.10. Controlling agent without parameters is shown in Figure 4.11, Controlling agent with parameters is shown in Figure 4.12.

```

$ http GET http://microgrid.ntu.edu.sg:8888/bic/0
HTTP/1.1 200 OK
Content-Length: 841
Content-Type: application/json
Date: Tue, 18 Oct 2016 14:08:10 GMT
Etag: "9cb30483de8ebf87d9245e7b713e965736cd1e29"
Server: TornadoServer/4.4.2

{
  "AC_Reference_frequency": 0.12,
  "AC_Reference_voltage": 0.5,
  "AC_VQ_droop_coefficient": 1.8,
  "AC_bus_Frequency": 0.05,
  "AC_bus_current_A": 0.24,
  "AC_bus_current_B": 0.18,
  "AC_bus_current_C": 0.14,
  "AC_bus_voltage_phase_A": 1.6,
  "AC bus voltage phase B": 0.5.

```

Figure 4.10: Reading Status of Agent by RESTful API

```

# longqi @ LQMacPro in ~ [22:14:37]
$ http PUT http://microgrid.ntu.edu.sg:8888/battery/0 cmd=TURN_OFF
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: text/html; charset=UTF-8
Date: Tue, 18 Oct 2016 14:14:54 GMT
Server: TornadoServer/4.4.2

```

Figure 4.11: Controlling Agent Without Parameters by RESTful API

```

# longqi @ LQMacPro in ~ [21:51:18]
$ http PUT http://microgrid.ntu.edu.sg:8888/battery/0 cmd=TURN_ON_POWER_DISPATCH reference_power=3500
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: text/html; charset=UTF-8
Date: Tue, 18 Oct 2016 13:54:04 GMT
Server: TornadoServer/4.4.1

```

Figure 4.12: Controlling Agent With Parameters by RESTful API

4.2.2.2 IEC 61850 API

Although IEC 61850 is officially described as a communication structure for substation and feeder equipment of a power utility control system, its most useful contribution is the definition of an object model for all substation objects, such as relay, converter and so on. IEC 61850 defines Intelligent Electronic Device (IED) as "device incorporating one or more processors with the capability to execute application functions, store data locally in a memory and exchange data with other IEDs (sources or sinks) over a digital link". Considering the modular microgrid's functionalities, one single microgrid can be treated as one IED which contains multiple Logical Node (LN)s.

To take advantage of the standard IEC 61850 and achieve the interoperability of two or more IEDs from the same vendor, or from different vendors, and to exchange information and execute specified tasks based on the information, all the field power devices need to be decompressed to proper LNs [14]. Based on the hardware configuration in the microgrid module as shown in Figure 2.4, the field devices can be grouped into the following categories: converter, sensor, switchgear and controller.

Most of the models of the above devices can be found in IEC 61850-5 [14], IEC 61850-7-410 [15] and IEC 61850-7-420 [2] directly but nowadays more and more Distributed Energy Resources (DERs) as shown in Figure 4.13 are used in microgrid. IEC 61850 does not cover all the power devices so it allows users to extend existing LN or define new LN based on the device requirements by following basic extension rules as shown in Figure 4.14.

Two methods are suggested to solve the problem. The first approach is to add or subtract some data of an existing logic node to meet the requirements. The other way is to define a completely new LN and use the predefined standard data through the existing LNs. The second method is selected to model the bidirectional converter including the functions that the rectifier and the inverter perform because no LNs can be applied for a bidirectional converter directly.

Figure 4.13: Logical Devices and Logical Nodes for DER Overview [2]

Figure 4.14: Basic Rules

4.2.2.3 Bidirectional Converter Modeling

The bidirectional converter plays a significant role of power balancing between AC and DC bus in the hybrid AC/DC microgrid module and it's also a relatively new device so IEC 61850 doesn't have a related definition. To make use of IEC61850, modeling is the most important step so we modeled the bidirectional converter in this section. A generic structure of a bidirectional converter in a microgrid module is shown in Figure 4.15. As the name suggests, a bidirectional converter is a bidirectional energy flow device which can transfer AC power from the utility grid or diesel generator to the DC bus to feed DC loads, and vice versa. ZRCT(rectifiers) and ZINV(inverters) are standard logic nodes in IEC 61850-7-420. Both of them can only partly cover the functions of a bidirectional converter. As the rules in Figure 4.14 suggests, the standard LN should be considered first, so we used ZRCT and ZINV as the basis to design the mapping model as shown in Figure 4.15.

Figure 4.15: LN for Bidirectional Converter

The list below is the description of different logical nodes in Figure 4.15.

- Logic node ZBIC defines the characteristics of the bidirectional converters used in hybrid AC/DC microgrid module. This LN is described in detail in the appendix.
- XCBR is the logic node used for abstracting the internal protection relays in the converter.
- CSWI works with XCBR and handles all switchgear operations from the operators and from related automatics.
- CPOW works with CSWI and provides all the functionalities to turn on or off a circuit breaker at a certain point of time.
- MMXU and MMDC get values from CTs and VTs and then calculate the intermediate variables like r.m.s. values or power flows.
- AVCO automatically controls the voltage of a bus bar within a specific range independent of the means used.
- ARCO automatically controls the reactive power flow in a substation within a specific range using capacitors and/or reactances.
- PTOC/PTOV is a function that operates when the input current/voltage exceeds a predetermined value, and in which the input current/voltage and operating time are inversely related through a substantial portion of the performance range.
- PTUV is a function that operates when its input voltage is less than a predetermined value.
- CCGR is used to control the cooling equipment.
- LNN0 represents the logic device specific information, such as the device name and health.

The bidirectional converter model we designed is shown in appendix B. ZBIC is inherited from the logical-node class. Z refers to the LN group of a further power system equipment. BIC

stands for bidirectional. The profile of the new logical node is listed in Appendix B. A detailed description is listed below:

- Add more device types based on the characteristics of a bidirectional converter.
- Change *OutFilTyp* to *FilTyp* because there is no fixed output and input for a bidirectional converter.
- Power direction and operation mode setting.
- Corresponding setpoint for different working modes.
- Limitation setting of AC/DC on both sides.

The other devices like the storage system and the photovoltaic system, can use the existing predefined model in IEC 61850-7-420 [2]. With active API, different modules with low coupling in the microgrid can work together. With passive API, multiple microgrid modules can be integrated as a scalable Power Park.

Chapter 5

Communication Network

5.1 Overview

The operational and commercial demands of electric utilities require a high-performance data communication network. Therefore, the design of a network architecture is crucial for the system performance. It's the same situation for the microgrid module. The agent model needs to synchronize with the corresponding field device which needs to be controlled by commands from the agent API. The different modules of microgrid OS modules also need to work as a whole.

Based on the hierarchical control [16], we designed a two-layer communication network for the microgrid module. The overview of the microgrid Power Park communication network is illustrated in Figure 5.1. There is one BBB as the central controller for the Power Park to provide high level information and manage several microgrids in a particular area. One BBB in every microgrid as the central controller is developed with a human machine interface, a secondary control and communication interfaces with field devices. The communication network is divided into two layers: FAN and PAN.

5.2 Field Area Network

FAN is the most important part to make the individual microgrid fully functional. The critical messages are transmitted to BBB on the FAN and BBB sends commands to every DSP.

Figure 5.1: Communication Network

All the DSPs are designed to be dynamically configurable with the Plug-N-Play feature. DSPs and BBB need to exchange properties and real-time information to meet the requirements of state synchronization.

5.2.1 CAN bus and OSI model

The OSI model is a conceptual model that characterizes and standardizes the communication functions of a telecommunication or computing system regardless of their underlying internal structure and technology. It is used widely as the reference of communication design. The seven layers are Physical, Data link, Network, Transport, Session, Presentation and Application.

The data flow following the OSI model is shown in Figure 5.2. The data that needs to be transmitted is ready in the transmitting node, flows down over the physical layer and up through to the receiving node. In a complete transmission following OSI, every layer adds a header to the data that becomes the new data for the next layer. This process in the transmitting device is called encapsulation as shown in Figure 5.3. The combined encapsulated new data

is then transmitted and received. The receiving device reverses the process, de-encapsulating or exposing the data at each layer under the instruction of the header information. Finally the receiving device gets the data from the transmitting device. The process needs to be done at least once for every transmission. OSI model can prevent changes in one layer from affecting other layers, and divide network communication processes into smaller components that makes software development, design, and troubleshooting easier. OSI model provides several services such as *Data Segmentation*, *Packet Acknowledgment* and so on, but these encapsulation and de-encapsulation operations take too many computation resources of a DSP which needs to take care of the real-time control of power electronic components.

Figure 5.2: OSI Data Flow

A lot of industrial communication protocols only follow one or several layers rather than the completely conceptual OSI model. Actually most of the industrial protocols only have one physical layer such as RS485, RS232 or LIN bus.

Encapsulation

Each layer introduces its own header and overhead

Figure 5.3: Encapsulation

CAN bus [17] is a vehicle communication standard designed to allow micro-controllers and devices to communicate with each other in applications without a host computer. It is a message-based protocol, designed originally for multiplex electrical wirings within automobiles, but is also used in many other contexts [18].

Figure 5.4: CAN Bus in OSI Model

Compared to the other communication protocols, CAN bus has a big advantage as shown in Figure 5.4. CAN bus has a physical layer and a data link layer in the OSI model that makes it easy to use and simplifies coding work and saves a lot of computation resources.

The functionality of the physical layer and data link layer in CAN is listed in Figure 5.5.

DSP on the converter mainly focuses on the hard real-time control on the power electronics so the FAN must be economical in the computing resource usage. It means that DSP can get correct messages from other communication nodes without too much calculation. RS485 and CAN are popular standards in the field bus system and both of them are equipped with DSP.

Figure 5.5: Functions in CAN Bus Layers

But compared with RS485, CAN bus is a network with independent controllers and a serial communication protocol that efficiently supports distributed real-time control with a very high level of security. It is capable of avoiding data collisions, detecting failures in the transmitted data, automatic repetition of disturbed messages and ensuring data consistency over all nodes in a network. All of these advantages can save valuable time for the real-time control as well as provide a high-speed and reliable field communication [19].

The data link layer ensures that all the packets are passed on free of errors. The three main functions of the data link layer are to deal with transmission errors, to regulate the flow of data, and to provide a well-defined interface to the upper layer.

5.2.2 Application Layer Design

To take advantage of the CAN bus protocol, we designed an application layer of CAN bus that simplifies the subsequent coding work and makes the communication system scalable and easy to reconfigure. There are some existing application protocols like CANopen or DeviceNet, but they are relatively too heavy in our case as shown in Figure 2.3. And some requirements

can not be met because of the protocol's self-limitation. There are less than ten devices in one microgrid module or in one CAN bus network. Less than 100 variables need to be sent from DSP to the central controller. Because there is a three-phase AC bus in the microgrid, a lot of data is grouped in threes. The three phases of the AC are closely related so the data about the three phases is better to be grouped and sent at the same time.

A basic complete CAN bus 2.0 message frame mainly includes *Identifier (ID) field*(green), *Data Length Code (DLC) field*(yellow) and *data field*(red) as shown in Figure 5.6.

Figure 5.6: CAN Bus 2.0A Frame

The ID field is 11 bits in CAN 2.0A and 29 bits in CAN 2.0B. The data field is 0-64 bits (0-8 bytes) which is indicated by the DLC field in bytes.

Figure 5.7: CAN ISO11898-2 High-Speed Network [3]

CAN is a multi-master serial bus standard for connecting two or more nodes in a CAN communication network. CAN 2.0 standard allows bit rates up to 1 Mbit/s at network cable lengths below 40 meters. There are two types of CAN bus connection for CAN network: ISO 11898-2 as shown in Figure 5.7 and ISO 11898-3 as shown in Figure 5.8. The former uses

Figure 5.8: CAN ISO11898-3 Low-Speed Fault Tolerant Network [4]

a linear bus terminated at each end with $120\ \Omega$ resistors. Considering the small-size module that all the communication nodes(BBB and DSP) are located in, we use the high-speed CAN network configuration in the microgrid module as shown in Figure 5.9.

Figure 5.9: CAN Bus Connection Microgrid Module

The CAN bus module in DSP has the functions of bit timing and synchronization, message framing, arbitration, acknowledgment, error detection and signaling, and fault confinement. Besides these functions in the transfer layer, *Message Filtering* and *Message & Status Handling* are also covered by the CAN bus module. The proper filter setting and exception handling functions are designed in the CAN bus module to provide a stable and high-speed

communication system.

Since the central controller BBB needs to receive all the messages from the converter controller DSP so we set BBB open for all messages on the bus.

DSP has 32 mailboxes which can be set at receiving mode or transmitting mode, and most of them are set at receiving mode and the rest are at transmitting mode since much more data needs to be sent to BBB compared to commands from BBB to DSP.

As a bus communication system, every communication node can receive any message from the sending node but itself. CAN bus module provides a message filtering service that only messages with desired IDs can be accepted automatically without the involvement of the application layer so we set the corresponding acceptance mask for every mailbox to reduce time consumption of DSP. One more function of the message ID is the priority mechanism deciding which message will be transmitted when collision happens. This makes CAN bus very suitable for a real-time prioritized communication system. The message with higher priority has no delay during the transmission. The low priority message will stop transmit and automatically attempt to re-transmit six bit clocks after the end of the dominant message. The CAN bus node that transmits the first one loses arbitration. Since the identifier is transmitted by all nodes at the start of the CAN frame, the node with the lowest identifier transmits more zeros at the start of the frame, and that is the node that wins the arbitration or has the highest priority. Based on the priority mechanism we set the message ID accordingly.

5.2.2.1 Protocol

The CAN bus message frame consisted of ID and data from the application layer's view. All the messages being exchanged between the central controller and the converter controller DSP are well predefined.

To utilize the CAN bandwidth, and make sure that the critical group data can be sent at the same time, One data frame consists of two parts as shown in Figure 5.10.

While the system is running, the central controller needs to collect two types of data:

- *Binary Data.* Field Device Switch Status, used for on/off status, such as relay and error

Figure 5.10: CAN Bus Frame Protocol

type. Normally one bit is for one status as shown in Figure 5.11.

- *Integer Data.* Converter Running Parameters, used for the measured value or current setting, such as the DC bus voltage (scale 0.1), reference AC bus voltage (scale 0.1) and AC bus frequency (scale 0.01).

So every CAN bus message consists of $8 \times 2 \times 3 = 48$ bits of data and an 8-bit index which is used to identify the data. The data type can be 16-bit signed integer ranging from 32,768 to 32,767 or 16-bit unsigned integer ranging from 0 to 65535. To have a high accuracy, a scale value of 0.1 or 0.01 is predefined for every value being transferred. The real value range is shown in Table 5.1.

Table 5.1: Value Range in 16-bit Integer

Scale	Signed 16-bit Integer	Unsigned 16-bit Integer
1	-32768 $\rightarrow$ 32768	0 $\rightarrow$ 65535
0.1	-3276.8 $\rightarrow$ 3276.8	0 $\rightarrow$ 6553.5
0.01	-327.68 $\rightarrow$ 327.68	0 $\rightarrow$ 655.35

Figure 5.11: Binary Data in CAN bus

5.2.2.2 Scalability

Scalability is a critical concern for the communication system. A typical conversation is shown in Figure 5.12. Module 1 sends some data to module 2 which takes time t_1 . Module 2 takes time t_2 to process and send commands back which takes time t_3 . Normally t_1 and t_2 have the same value which is from the experiment shown in Figure 5.13.

Based on the time requirement shown in Figure 3.1, the minimal response time is much less than 1 second. So we can have $N = 1 \times 10^6 / 212 \approx 4717$ conversations. Actually most of the messages are unidirectional data transmitting from DSP to BBB. A complete conversation only happen during a RTC task execution or fault coping. Since the microgrid module consists of only 5 DSP devices and there are less than 20 messages for each device, the communication design leaves wide extending space for scalability.

Figure 5.12: Data Transmission in CAN bus

Figure 5.13: Waveform of CAN Message Conversation

5.3 Park Area Network

We designed *PAN* which provides communication between microgrid modules in the Power Park so it can cross over the whole Power Park and connect every single microgrid module as shown in Figure 5.1. It is a high-bandwidth backbone communication network that handles long-distance data transmission with advanced monitoring and sensing applications.

PAN provides a two-way communication network for automation and monitoring. With IEC61850 and RESTful API adapted on the central controller, multiple microgrid modules can be easily interconnected as a Power Park or integrated with utility grid. PAN is to Power Park as Wide Area Network (WAN) is to Smart Grid [20]. The only difference is that PAN is smaller so PAN can also use the technology that WAN uses. Based on the local operation conditions such as ESS State of Charge (SoC) and the open market price, the microgrid can automatically adjust the control strategy to fully exploit the use of renewable energy systems.

Depending on the type of renewable energy, the environment where microgrids are deployed and other restrictions, microgrid modules can be located far from each other. To get an economical remote communication, we utilize the Internet Protocol based communication, as shown in Figure 5.14. BBB running Linux as a full feature computer can be easily expanded and get access to the Internet by built-in 100M *Ethernet*, *Wi-Fi* and *cellular network* (3G/4G).

The UEMS and users' devices like smart phones and laptops, same as every microgrid module, are connected to Internet. We set up a Secure Sockets Layer (SSL) Virtual Private Network (VPN) server on UEMS and a VPN client on the microgrid module so the API can be accessed inside the VPN to ensure the safety of the microgrid module. On the other hand, SSL and VPN are located on the Data Link layer and Network layer of OSI, so the PAN setup can be independent with the microgrid OS development.

The two-layer communication is established with FAN and PAN so the heavy controlling work can be distributed evenly. By integrating the information and communication technology, the microgrid module can be achieved as a sophisticated system so RESs can be fully utilized and the system energy usage can be more efficient.

Figure 5.14: PAN of the communication network

Chapter 6

Conclusion and Future Work

In this chapter, the main contributions of this research project are summarized and the conclusions of the thesis are presented. Recommendations for the future work of this research project are also proposed.

6.1 Conclusion

In recent years, microgrid is becoming more and more popular as it has been regarded as a viable bottom-up solution platform for the ultimate smart grid whose goal is dramatically boosting efficiencies, thereby minimizing costs since more and more distributed renewable energy sources are being used everywhere.

The main contribution of this thesis is to provide a single board computer based low-cost and scalable platform for a modular microgrid which can be tied to utility grid or form a modular microgrid based Power Park. Most of the existing SCADA or EMS are mainly designed for substation and also need Windows OS, related exclusive hardwares and expensive licensees. This new platform using OS independent web technology leaves the heavy GUI rendering work to any client such as a smart phone, tablet and laptop so the single board computer can focus on the control and EMS functions blocks. The well-designed API including active API and passive API decouple the minimum system, GUI and long-term scheduling so that real-time control, GUI and EMS function blocks like ED and UC can work and develop independently. This platform is also the main operating system for a 10-million R&D project *Smart Multi-Energy*

System (SMES) which is a joint-proposal between NTU and JTC under the Sustainable Energy Research & Technology Demonstrator (SERTD) initiative and was launched several months ago.

6.2 Future Work

The FAN network is now using CAN bus as the field communication bus, but for a microgrid, more commercial devices can also be connected. We should have a Hardware Abstraction Layer (HAL) for different hardwares and different protocols just like HAL of OS so the platform can have a plug-n-play feature instead of rewriting the underlying code. Wireless is also another possible scene that may happen in the future in the SMES.

APPENDIX A

Author's Publication

A.1 Conference Papers

- **Longqi Zhang**, Jianfang Xiao, Xuewei Pan and Peng Wang, “Design and Implementation of Communication Network for Modular Microgrid based Power Park”, *IEEE PES Innovative Smart Grid Technologies Asia Conference (ISGT)*. Nov. 2016
- **Jianfang Xiao**, and Longqi Zhang, “Distributed Coordination for Economical Operation of Power Park with Multiple Microgrids”, *Asian Conference on Energy, Power and Transportation Electrification (ACEPT)*. Oct. 2016

APPENDIX B

Logic Node of Bidirectional Converter

ZBIC class					
Data object name	Common data class	Explanation		T	M/O/C
LNName		Shall be inherited from logical-node class (see IEC 61850-7-2)			
Data					
System logic node data					
		LN shall inherit all mandatory data from common logical node class			M
		Data from LLN0 may optionally be used			O
WRtg	ASG	Maximum power rating			M
VarRtg	ASG	Maximum var rating: var			M
SwTyp	ENG	Switch Type:		O	
		Value	Explanation		
		0	Not application / Unknown		
		1	Field effect transistor		
		2	Insulated gate bipolar transistor		
		3	Thyristor		
		4	Gate tyen off thyristor		
		99	Other		
CoolTyp	ENG	Type of cooling method:		O	
		Value	Explanation		
		0	Not applicable / Unknown		
		1	Passive air cooling (heatsink)		
		2	Forced air cooling (fan + heatsink)		
		3	Fluid cooling (water)		
		4	Heat pipe		
		99	Other		
IsoTyp	ENG	Type of isolation:		M	
		Value	Explanation		
		0	Not applicable / Unknown		
		1	Power frequency transformer isolated		
		2	Hi frequency transformer isolated		
		3	Non-isolated, grounded		
		4	Non-isolated, isolated DC source		
		99	Other		
VregTyp	ENG	Type of voltage regulation:		M	
		Value	Explanation		
		0	Not applicable / Unknown		
		1	Regulated output: fixed voltage		
		2	Regulated output: variable voltage		
		3	Filtered output: load dependent		
		4	Unregulated and unfiltered		
		99	Other		
		Conversion type:			
		Value	Explanation		

ConvTyp	ENG	0	Not applicable / Unknown	0
		1	AC / DC or DC / AC	
		2	AC / AC / DC or DC / AC / AC	
		3	AC / DC / DC or DC / DC / AC	
		99	Other	
FilTyp	ENG	Filter type:		0
		Value	Explanation	
		0	Not applicable / Unknown	
		1	None	
		2	Series filter (L)	
		3	Parallel filter (LC)	
		99	Other	
InWavTyp	ENG	Input waveform conditioning type:		0
		Value	Explanation	
		0	Not applicable / Unknown	
		1	None	
		2	EMI filter	
		3	Line filter	
		4	EMI/Line filter	
		99	Other	
PQVLim	CSG	P-Q-V set of limiting curves		0
SwHz	ASG	Nominal frequency of switvhing		0
GridMod	ENG	Power system connect modes to the power grid:		0
		Value	Explanation	
		0	Not applicable / Unknown	
		1	Current-source inverter (CSI)	
		2	Voltage-controlled voltage-source inverter (VC-VSI)	
		3	Current-controlled voltage-source inverter (CC-VSI)	
Status Information				
GridModSt	ENG	Power system connect modes to the power grid:		0
		Value	Explanation	
		0	Not applicable / Unknown	
		1	Disconnected	
		2	Power not delivered	
		3	Power delivered	
99	Other			
Stdby	SPS	converter stand-by status True: stand-by active		0
CurLev	SPS	DC current level available for operation True: sufficient current		0
WDirSt	ENG	Power transferring direction:		0
		Value	Explanation	
		0	AC to DC	
		1	DC to AC	

ACTyp	ENG	AC system type:		M
		Value	Explanation	
		0	Single Phase	
		1	Two Phase	
		2	Three Phase	
Settings				
PQVLimSet	CSG	Active curve characteristic curve for PQV limit		M
OptMod	ENG	Operating mode		M
		Value	Explanation	
		0	Recitifier mode	
		1	Inverter mode	
		2	Power dispatch mode	
DVSet	ASG	DC voltage setpoint (only valid when OptMod is Rectifier mode)		O
AVSet	ASG	AC voltage setpoint (only valid when OptMod is Inverter mode)		O
HzSet	ASG	AC frequency setpoint (only valid when OptMod is Inverter mode)		O
WSet	ASG	Operating power setpoint (only valid when OptMod is Power dispatch mode):		O
		Value	Explanation	
		Positive	AC to DC	
		Negative	DC to AC	
VarSet	ASG	Operating reactive power setpoint (only valid when OptMod is Power dispatch mode):		O
		Value	Explanation	
		Positive	AC to DC	
		Negative	DC to AC	
PFSet	ASG	Power factor setpoint as angle (only valid when OptMod is Power dispatch mode):		O
		Value	Explanation	
		Positive	AC to DC	
		Negative	DC to AC	
AALim	ASG	AC side current limit		O
DALim	ASG	DC side current limit		O
AVLim	ASG	AC side voltage limit		O
DVLim	ASG	DC side voltage limit		O
PhACnfg	ENG	AC side Phase A feed configuration:		O
		Value	Explanation	
		0	Not applicable / Unknown	
		1	Feeding from N to A	
		2	Feeding from N to B	
		3	Feeding from N to C	
		4	Feeding from A to B	
		5	Feeding from A to C	
		6	Feeding from B to A	
		7	Feeding from B to C	
8	Feeding from C to A			

		9	Feeding from C to B	
		99	Other	
PhBCnfg	ENG	AC side Phase B feed configuration: see PhACnfg for enumerated		0
PhCCnfg	ENG	AC side Phase C feed configuration: see PhACnfg for enumerated		0
Measured values				
HeatSinkTmp	MV	Heat sink temperature: Alarm if over max		0
EnclTmp	MV	Enclosure temperature		0
AmbAirTmp	MV	Ambient outside air temperature		0
FanSpdVal	MV	Measured fan speed: Tach or vane		0

BIBLIOGRAPHY

- [1] S. Roostaei. Introduction to iec 61850. [Online]. Available: <https://protectionview.blogspot.sg/2016/04/introduction-to-iec-61850.html>
- [2] I. E. Commission. Communication networks and systems for power utility automation part 7-420: Basic communication structure distributed energy resources logical nodes. [Online]. Available: <http://www.iec.ch/smartgrid/standards/>
- [3] I. O. for Standardization. Road vehicles – controller area network (can) – part 1: Data link layer and physical signalling. [Online]. Available: https://en.wikipedia.org/wiki/CAN_bus
- [4] ———. Road vehicles – controller area network (can) – part 1: Data link layer and physical signalling. [Online]. Available: https://en.wikipedia.org/wiki/CAN_bus
- [5] Solarpower europe market report 2015. [Online]. Available: <http://www.solarpowereurope.org/home/>
- [6] X. Liu, P. Wang, and P. C. Loh, “A hybrid ac/dc micro-grid,” in *IPEC, 2010 Conference Proceedings*, Oct 2010, pp. 746–751.
- [7] Etap ems. [Online]. Available: <http://etap.com/energy-management-system/energy-management-system-software.htm>
- [8] Poweron reliance. [Online]. Available: <http://www.gegridsolutions.com/uos/ems.htm>
- [9] Schneider electric. [Online]. Available: <http://www.schneider-electric.com/>
- [10] P. Jintagasonwit, “Implementation techniques of the ied and network monitoring system in iec61850 sa,” in *Transmission and Distribution Conference and Exposition (T&D), 2012 IEEE PES*. IEEE, 2012, pp. 1–5.
- [11] L. Richardson and S. Ruby, *RESTful web services*. ” O’Reilly Media, Inc.”, 2008.
- [12] J. Xiao, P. Wang, L. Setyawan, and Q. Xu, “Multi-level energy management system for real-time scheduling of dc microgrids with multiple slack terminals,” *IEEE Transactions on Energy Conversion*, vol. 31, no. 1, pp. 392–400, 2016.
- [13] R. Alur and D. L. Dill, “A theory of timed automata,” *Theoretical computer science*, vol. 126, no. 2, pp. 183–235, 1994.
- [14] I. E. Commission. Communication networks and systems for power utility automation part 5: Communication requirements for functions and device models. [Online]. Available: <http://www.iec.ch/smartgrid/standards/>

- [15] ——. Communication networks and systems for power utility automation part 7-410: Basic communication structure hydroelectric power plants communication for monitoring and control. [Online]. Available: <http://www.iec.ch/smartgrid/standards/>
- [16] V. C. Gungor and F. C. Lambert, “A survey on communication networks for electric system automation,” *Computer Networks*, vol. 50, no. 7, pp. 877–897, 2006.
- [17] L. Jianxin, G. Xuefeng, and T. Ping, “The analysis and test of real-time performance for time-triggered can bus,” in *2008 IEEE International Conference on Automation and Logistics*. IEEE, 2008, pp. 3005–3009.
- [18] X. Gao and L.-s. Li, “Analysis and research of real time ability of message transmission in can bus,” in *Control, Automation and Systems Engineering (CASE), 2011 International Conference on*. IEEE, 2011, pp. 1–3.
- [19] L. Ding, Y. Shen, and Y. Zhou, “Real time performance analysis and evaluation of can bus with an extended petri net model,” in *Instrumentation and Measurement Technology Conference, 2009. I2MTC'09. IEEE*. IEEE, 2009, pp. 1081–1084.
- [20] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, “A survey on smart grid communication infrastructures: Motivations, requirements and challenges,” *IEEE communications surveys & tutorials*, vol. 15, no. 1, pp. 5–20, 2013.