Designing Modeling Tool for Embedded Systems and Sensor Networks

By

Shah Stuti V.

Roll No: 09MCE029

Department of Computer Science and Engineering

Institute of Technology

Nirma University

Ahmedabad-382481

May,2011

Designing Modeling Tool for Embedded Systems and Sensor Networks

Major Project

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Computer Science and Engineering

By

Shah Stuti V.

Roll No: 09MCE029

Guided By

Mrs. Ekata Mehul

Dr. S.N.Pradhan

Department of Computer Science and Engineering

Institute of Technology

Nirma University

Ahmedabad-382481

May,2011

Declaration

This is to certify that

i) The thesis comprises my original work towards the degree of Master of Technology in Computer Science and Engineering at Nirma University and has not been submitted elsewhere for a degree.

ii) Due acknowledgement has been made in the text to all other material used.

Shah Stuti V.

09MCE029

Certificate

This is to certify that the Major Project entitled "Designing Modeling Tool for Embedded Systems and Sensor Networks" submitted by Shah Stuti V. (09MCE029), towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering of Nirma University, Ahmedabad is the record of work carried out by her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project part, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Mrs. Ekata Mehul
External Guide,
Training Manager,
eInfochips Ltd.,
Ahmedabad.

Dr.S.N.Pradhan
Internal Guide and Professor,
Department of Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Prof.D.J.Patel
Professor and Head,
Department of Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Dr. K Kotecha
Director,
Institute of Technology,
Nirma University, Ahmedabad.

Abstract

With the complexity of developing the Embedded Systems, modeling them before their construction is very essential. Hence, modeling tools are increasingly being used as the primary development environment for creating, designing and simulating Embedded Systems. Sensor Networks also posses real time constraints modeling Sensor Networks is essential before implementation. A survey has been conducted on various open source modeling tools and their features are highlighted. Approach adopted for finding suitable modeling tool is to deal with finding essential features for the designing Embedded Systems and Sensor Networks and comparing them with features provided by different modeling tools.Tables showing comparison for different tools are presented in chapter 2.After analysis it was found that Ptolemy is one of the best available open source modeling tool for designing Embedded Systems.Also Ptolemy provides various tools including Sensor Networks development. Prototyping an embedded system with Ptolemy fulfils all the essential requirements for designing an Embedded Systems and Sensor Networks.Old Age Assistive Device is taken as reference case study to enhance Ptolemy as Ptolemy is open source modeling tool. Architecture study of Ptolemy is done for inclusion of components in to Ptolemy library. Old Age Assistive Device is simulated in Ptolemy to find out missing features with Ptolemy. Library of sensors for Old Age Assistive Device are added to Ptolemy user library so the similar types of systems can be implemented in future without any major efforts. Thus, the purpose of designing modeling tool is served. Old Age Assistive Device is implemented on hardware using model based approach adopted in Ptolemy. Thus, Modeling reduces designing time of any embedded system and also give proof of concept at much earlier stage reducing errors at later stage.

Acknowledgement

It gives me great pleasure in expressing thanks and profound gratitude to **Mrs. Ekata Mehul,** Training Manager, eInfochips, Ahmedabad for her valuable guidance and continual encouragement throughout the Major project. I heartily thankful to her for her time to time suggestions and the clarity of the concepts of the topic that helped me a lot during this study.

I would like to extend my gratitude to **Dr.S.N.Pradhan,** Professor & M.Tech Coordinator, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad for fruitful discussions and valuable suggestions during meetings and for their encouragement.

I would like to thanks **Dr.Ketan Kotecha,** Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad for providing basic infrastructure and healthy research environment.

I am thankful to the almighty who blessed me with the zeal to work hard.

Last, but not the least, no words are enough to acknowledge constant support and sacrifices of my family members and friends because of whom I am able to complete the dissertation work successfully.

Shah Stuti V.

09MCE029

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The underlying goal of this project is to design a modeling tool for Embedded Systems and Sensor Networks. This document elaborates the methodology adapted for designing modeling tool. First of all survey has been conducted for finding best suitable modeling tool available for designing Embedded Systems as well as Sensor Networks. For survey, features required for designing any Embedded Systems and Sensor Networks are collected and comparison table has been made for various available open source modeling tools with respect to that feature. Concluding Ptolemy as best suitable open source modeling tool available for designing Embedded Systems and Sensor Networks, simulation of a Old Age Assistive Device is done on Ptolemy. Some of the missing features are added to Ptolemy library as Ptolemy is open source modeling tool. Old Age Assistive Device is also implemented on hardware with the partial help of model simulated in Ptolemy.Model based approach for designing and implementation reduces time requires designing and implementing system.

1.1 Objective of the Work

There are two main objectives of this thesis. One is designing modeling tool for Embedded Systems and Sensor Networks and second objective is to implement Old Age Assistive Device using model based approach that can be used for old age people

which in turn will serve society.

1.2 Motivation of the Work

The motivation behind doing this research is to use model based approach for design and implementation of any system. Enhancing open source modeling tool will increase the features available for simulating any system on that tool and implementation of system using model based approach will reduce the time require for design and simulating any system

1.3 Embedded Systems

An Embedded System is some combination of computer hardware and software, either fixed in capability or programmable, that is specifically designed for a particular function. Industrial machines, automobiles, medical equipment, cameras, household appliances, airplanes, vending machines and toys (as well as the more obvious cellular phone and PDA) are among the myriad possible hosts of an embedded system.

Figure 1.1: Embedded Systems

1.3.1 Applications of Embedded System

- Medical electronics technology.
- Communications applications.
- Electronics applications and consumer devices.
- Industrial automation and process control software.

1.4 Sensor Networks

Wireless sensor networks are a trend of the past few years, and they involve deploying a large number of small nodes. The nodes then sense environmental changes and report them to other nodes over a flexible network architecture. Sensor nodes are great for deployment in hostile environments or over large geographical areas.

1.4.1 Application of Sensor Networks

Sensor networks have been useful in a variety of domains. The primary domains at which sensor are deployed follow:

- Environmental observation.

  Sensor networks can be used to monitor environmental changes. An example could be water pollution detection in a lake that is located near a factory that uses chemical substances. Sensor nodes could be randomly deployed in unknown and hostile areas and relay the exact origin of a pollutant to a centralized authority to take appropriate measures to limit the spreading of pollution. Other examples include forest fire detection, air pollution and rainfall observation in agriculture.

- Military monitoring.

  Military uses sensor networks for battlefield surveillance; sensors could monitor vehicular traffic, track the position of the enemy or even safeguard the equipment of the side deploying sensors.

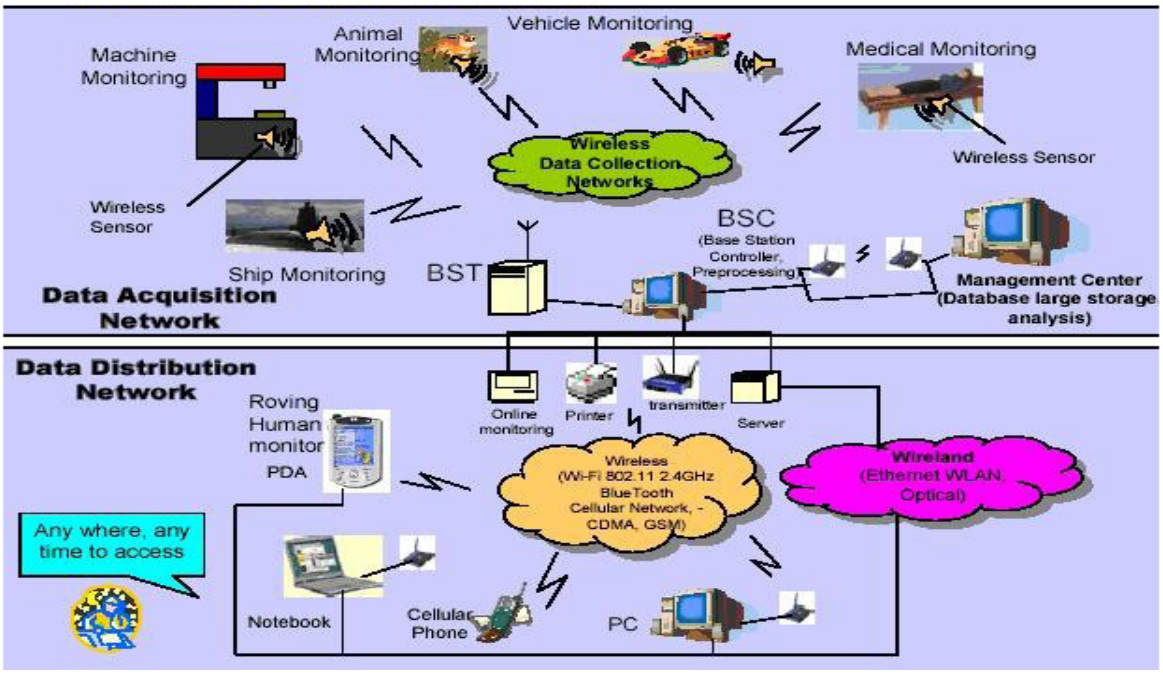

Figure 1.2: Sensor Networks

- Building monitoring.

  Sensors can also be used in large buildings or factories monitoring climate changes. Thermostats and temperature sensor nodes are deployed all over the buildings area. In addition, sensors could be used to monitor vibration that could damage the structure of a building.

- Healthcare

  Sensors can be used in biomedical applications to improve the quality of the provided care. Sensors are implanted in the human body to monitor medical problems like cancer and help patients maintain their health.

1.5 Modeling Tool

Modeling is the process of producing a model; a model is a representation of the construction and working of some system of interest. A model is similar to but simpler than the system it represents. One purpose of a model is to enable the analyst to predict the effect of changes to the system.On one hand, a model should be a close approximation to the real system and incorporate most of its salient features.While on the other hand, it should not be so complex that it is impossible to understand and experiment with it.A good model is a judicious tradeoff between realism and simplicity.Simulation practitioners recommend increasing the complexity of a model iteratively.An important issue in modeling is model validity. Model validation techniques include simulating the model under known input conditions and comparing model output with system output.

Generally, a model intended for a simulation study is a mathematical model developed with the help of simulation software. Mathematical model classifications include deterministic (input and output variables are fixed values) or stochastic (at least one of the input or output variables is probabilistic); static (time is not taken into account) or dynamic (time-varying interactions among variables are taken into account).Typically, simulation models are stochastic and dynamic.

1.6 Simulation

A simulation of a system is the operations performed on model of the system. The model can be reconfigured and experimented with; usually, this is impossible,too. expensive or impractical to do in the system it represents. The operation of the model can be studied, and hence, properties concerning the behavior of the actual system or its subsystem can be inferred. In its broadest sense, simulation is a tool to evaluate the performance of a system, existing or proposed, under different configurations of interest and over long periods of real time.

1.7 Thesis Organization

The rest of the thesis is organized as follows.

Chapter 2, *Literature Survey*, presents the literature survey done for finding suitable modeling tool for designing Embedded Systems and sensor networks.

Chapter 3,*Old Age Assistive Device*, the functionality of each and every mode is defined and hardware and software parts of the device is separated.

Chapter 4,*Designing of Old Age Assistive Device*, presents the designing of modes of operation of Old Age Assistive Device

chapter 5,*Simulation*, presents the methodology adopted for simulating Old Age Assistive Device on to Ptolemy. Sensors are simulated and added as part of Ptolemy library.

chapter 6, *Implementation*, This chapter presents the use of hardware for implementing Old Age Assistive Device. The model based approach is described.

Finally, in **chapter 7** concluding remarks and future work is presented.

Chapter 2

Literature Survey

This chapter elaborates the survey carried out for finding best suitable open source modeling tool available for modeling of Embedded System with Support of Sensor Networks. Features of surveyed modeling tools are listed. Features required for designing Embedded Systems and Sensor Networks are found out and comparison table is made comparing various modeling tools with reference to desired feature.

2.1 Ptolemy

- Java: Ptolemy II is implemented in Java, so it has all the advantages of Java.
- Platform independent - Today, Ptolemy II runs under Solaris and Windows NT.
- Threaded - Ptolemy II builds on the threading primitives in Java.
- Adaptation of modern type theories to block-diagram-level specification of systems.
- Timed extensions of communicating sequential processes (CSP) and process networks (PN).
- A modular approach to synchronous/reactive design.

- A mathematical framework for comparing models of computation and studying their interaction.
- A bounded execution policy for process networks.
- Formalization of computational models for dataflow.
- Managing regularity in dataflow graphs using higher-order functions.
- Automating the design of multidimensional multirate systems.
- Simulating and scheduling multidimensional multirate systems.
- Simulating heterogeneous systems.
- Modular hierarchical finite state machines with various concurrency models.
- Synthesizing embedded software from dataflow graphs.
- Parallel scheduling techniques.
- Optimizing inter processor communication in parallel implementations.
- Deriving fast algorithms for hardware/software partitioning of dataflow graphs.
- Managing tool invocations and data dependencies in design processes.
- Integrated heterogeneous design visualization.
- Use of programming language concepts such as semantics, type theories, reflection, and concurrency theories in system-level design of electronic systems.
- Focus on domain-specific modeling and design problems so the designer can focus on the problem, not the tools.
- Emphasis on understanding of systems, which is promoted by visual representations, executable models, and verification.

- Use of Java, design patterns, UML, and a modern software engineering practice adapted to the realities of academic research.
- Ptolemy supports SR (Synchronous Reactive)domain , CT (Continuous Time) , DT (Discrete Time), PN (Process Networks), SDK (Synchronous Data Flow), DDE (Discrete Data Flow), DE (Discrete Event)

2.2 Scilab

- Hundreds of mathematical functions.
- High level programming language.
- 2-D and 3-D graphics.
- Advanced data structures and user defined data types.
- hybrid dynamic systems modeler and simulator.

2-D and 3-D visualization

- Lines
- Pie charts
- Histograms
- Surfaces
- Animations
- Graphics export in many formats: GIF, BMP, JPEG, SVG, PDF...

Numerical Computation

- Linear algebra

- Sparse matrices
- Polynomials and rational functions
- Simulation: explicit and implicit systems of differential equations solvers
- Classic and robust control
- Differentiable and non-differentiable optimization

Data Analysis

- Interpolation, approximation
- Signal Processing
- Statistics

Extended Features

- Graphs and Networks
- Interface with Fortran, C, C++, Java
- Functions for calling Scilab from C, C++, Fortran and Java
- LabVIEW Gateway
- A large number of modules available via ATOMS

2.3 FreeMat

- environment for rapid engineering and scientific processing. It is similar to commercial systems such as MATLAB.
- Include features such as a codeless interface to external C/C++/FORTRAN code, parallel/distributed algorithm development (via MPI), and advanced volume and 3D visualization capabilities.

- N-dimensional array manipulation (by default, N is limited to 6)
- Support for 8,16, 32, and 64 bit integer types (signed and unsigned), 32 and 64 bit floating point types, and 64 and 128 bit complex types.
- Built in arithmetic for manipulation of all supported data types.
- Support for solving linear systems of equations via the divide operators.
- Eigenvalue and singular value decompositions
- Full control structure support (including, for, while, break, continue, etc.)
- 2D plotting and image display
- Heterogeneous array types (called "cell arrays" in MATLAB-speak) fully supported
- Full support for dynamic structure arrays.
- Arbitrary-size FFT support
- Pass-by-reference support (an IDL feature)
- Keyword support (an IDL feature)
- Codeless interface to external C/C++/FORTRAN code.
- Native Windows support.
- Native sparse Matrix support.
- Native support for Mac OS X (no X11 server required).
- Function pointers
- Classes, operator overloading
- 3D Plotting and visualization via OpenGL

- Parallel processing with MPI (in the FreeMat development versions)
- Handle-based graphics

2.4 Comparision of Open Source Modeling Tools

Feature	Ptolemy	Scilab	FreeMat
Deadline	Available	NA	NA
Duration	Available	NA	NA
Continuous Time	Available	Available	Available
Discrete Time	Available	Available	Still to find
Real Time Constraints	Still to find	NA	NA
Time Multitasking	Available	NA	NA
Delay	Available	Available	NA
Execution time	Available	Available	NA
Interrupt management	Available	NA	NA
Event generation (SR , RS ..)	Available	Available	NA
Multi rate	Available	NA	NA
Message passing	Available	Available	Available
Synchronization	Available	Available	Available
Transitions	Available	Available	Available
Initial conditions.	Available	Available	Available

Table I: Timing Related Features

Feature	Ptolemy	Scilab	FreeMat
Task Scheduling	Available	NA	NA
Parallel scheduling techniques	Available	NA	NA
Concurrency support in modeling	Available	NA	NA
Sequential process	Available	Available	Available

Table II: Scheduling Capabilities

Feature	Ptolemy	Scilab	FreeMat
UML modeling	NA	NA	NA
XML code	Available	NA	NA
Graphical modeling	Available	Available	NA
Hybrid System modeling	Available	NA	NA
Heterogeneous system modeling	Available	NA	NA
Hierarchical modeling	Available	NA	NA
Finite State Machine	Available	NA	NA
Domain Specific modeling	Available	NA	NA
Modeling with programming	Still to find	Available	Available
Design model by Formal methods	Still to find	NA	NA
Kahn Process Networks	Available	NA	NA
Actor Oriented Design	Available	NA	NA

Table III: Different Way Embedded Systems can be Designed

Feature	Ptolemy	Scilab	FreeMat
Memory	Available	Available	Available
Processor	Available	NA	NA
A/D converter	Available	NA	NA
D/A converter	Available	NA	NA
CODEC	Still to find	NA	NA
Sensors and Actuators	Available	NA	NA
Global Memory	Available	Available	Available
Relations with components	Available	Still to find	Still to find
Visual representations	Available	Available	Available
Ports for communication and parameters configuring	Available	Available	NA

Table IV: Basic Blocks of Embedded Systems

Feature	Ptolemy	Scilab	FreeMat
Synchronous data flow	Available	Available	NA
Asynchronous data flow	Available	Available	NA
Dynamic data flow	Available	NA	NA
Discrete data flow	Available	Available	NA
Continuous data flow	Available	Available	Available
Handling data dependencies	Available	Available	NA

Table V: Different Dataflow Required

Feature	Ptolemy	Scilab	FreeMat
Open source	Available	Available	Available
User	Available	NA	NA
Power	Available	Available	NA
Output	Available	Available	Available
Input	Available	Available	Available
3D visualization	Available	Available	Available
Plot options	Available	Available	Available
Mathematical expression evaluation capabilities	Available	Available	Available
Signal processing capabilities	Available	Available	Available
Ports for communication and configuration	Available	NA	NA
Basic Operations	Available	Available	Available
Communication interface between components	Available	NA	Still to find
QOS parameters	Still to find	Still to find	Still to find
Translatable (Different language support)	Available	Available	Available
Synchronization	Available	Available	Still to find
Communication interface between components	Available	Still to find	Still to find
Code generation	Available	Available	Available
Compiler	Available	Available	Available

Table VI: Basic Requirements

Feature	Ptolemy	Scilab	FreeMat
Validation	Available	Available	Available
Simulation	Available	Available	Available
Verification	NA	NA	NA
Functional testing	Still to find	Available	Available
Debugging of model	Available	Available	Available
Create instances (while simulating)	Available	Still to find	Still to find
Hardware Software co design	NA	NA	NA
Model transformation	Available	NA	NA
Reusable models	Still to find	NA	NA
Models for computation	Available	NA	NA
Fail over - Fail back	NA	NA	NA
Operations	Available	Available	Available

Table VII: Requirements for Checking Correctness of System

Feature	Ptolemy	Scilab	FreeMat
Memory	Available	Available	Available
Processor	Available	NA	NA
A/D converter	Available	NA	NA
D/A converter	Available	NA	NA
CODEC	Still to find	NA	NA
Sensors and Actuators	Available	NA	NA
Global Memory	Available	Available	Available
Relations with components	Available	Still to find	Still to find
Visual representations	Available	Available	Available
Ports for communication and parameters configuring	Available	Available	NA
Roll back (Recovery requirements)	Available	NA	NA

Table VIII: Basic Blocks of Embedded Systems

Feature	Ptolemy	Scilab	FreeMat
Network protocols	Available	NA	NA
MAC layer protocol	Still to find	NA	NA
Collision detection	Available	NA	NA
Distance between node	Available	NA	NA
Power factor	Available	NA	NA
Heterogeneous environment	Available	NA	NA
Acknowledgement mechanism	NA	NA	NA
Re transmission of packet if lost	Available	NA	NA
Detecting whether node is in coverage area	Available	NA	NA
No. of nodes	Available	NA	NA
hand over	Available	NA	NA
Bandwidth	Available	NA	NA
Communication channel used	Available	NA	NA
Support mobility (node moving here and there)	Available	NA	NA
Wireless support	Available	NA	NA
Ad hoc support	NA	NA	NA
Security	Available	NA	NA
Data storage (memory)	Available	NA	NA

Table IX: Features Required for Sensor Networks

2.5 Summary

This chapter summarize features required for designing Embedded Systems as well as Sensor Networks.After comparing various available open source modeling it was found out ptolemy as best available open source modeling tool for designing Embedded Systems. It provides most of the requirement for simulating Embedded Systems and Sensor Networks. It also provides code generation facility from model up to certain extent.

Chapter 3

Old Age Assistive Device

3.1 Idea of Product

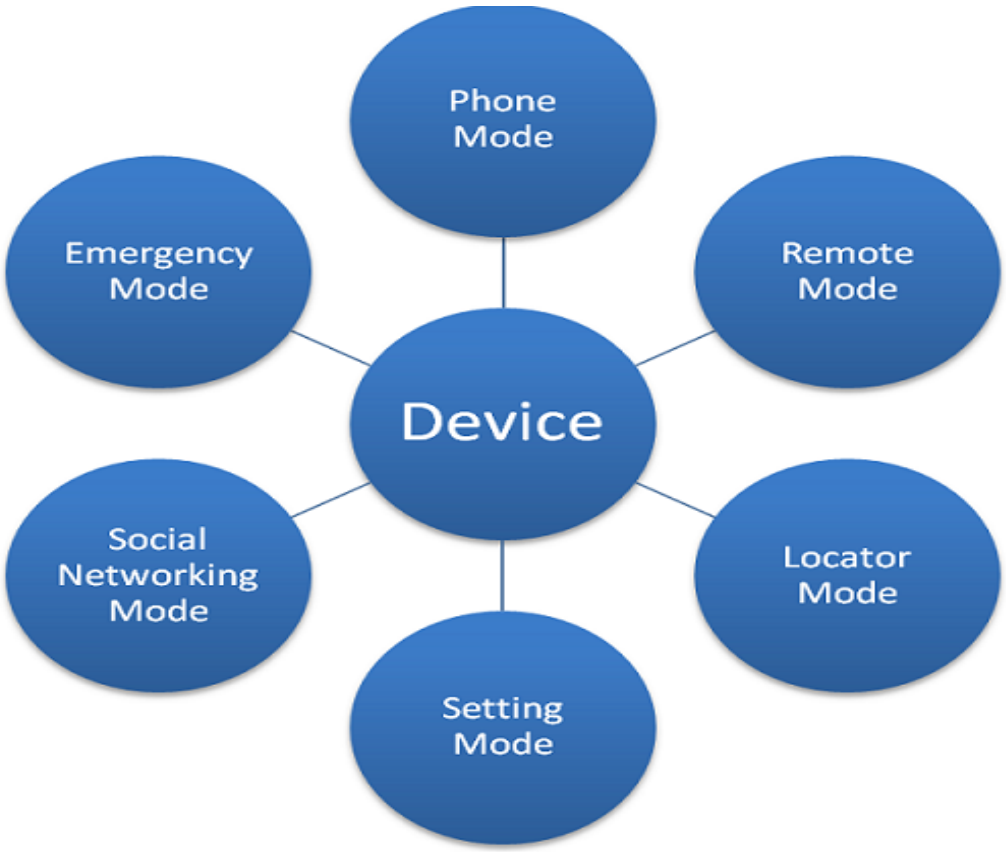

Figure 3.1: Old Age Assistive Device

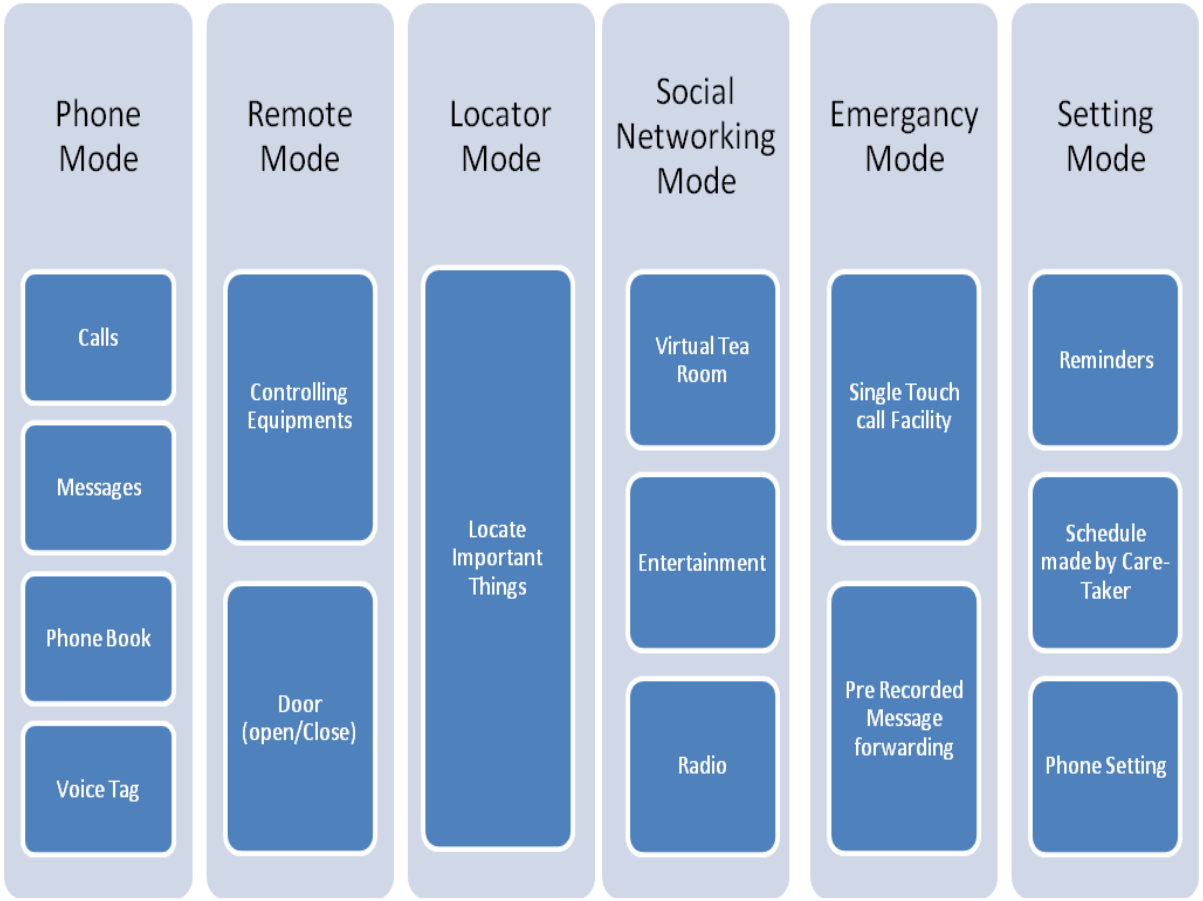

Figure 3.2: Mode Description

3.1.1 Mode Description

- **Phone Mode**

  It supports all features of mobile phone like calling or messaging someone. There is feature of voice tag which eliminates the need of searching the phonebook to

call anyone. They can call by just speaking his/her name. Voice tag is very useful for them because old one faces problem to type the mobile number or finding their names. So they can call easily without any problem. Touch screen is provided for ease of use. No need to press button to operate phone and its functionalities. By this mode, they can stay connected with loved ones, relatives, their friends or when they need to contact any other person for some work so they can easily communicate with them. So this feature can avoid dependency. This feature is support by Portable Controlling Device. This mode will require very costly chip set in implementation. For simulation Finite state Machine modeling is used.

- **Remote Mode**

  Remote mode provides old one easy operating of appliances. It supports the controlling of domestic equipment like T.V, AC, Fan, Light etc. This mode also support feature of open/close door which is very useful to them. If someone at door and press bell then his/her photo is taken by camera and displayed on wall mounted device. So, they can recognize them and open/close door whatever they want. This mode will help to reduce dependency on others because they can operate light/fan without any problem. Display will show position of light/fan and by just pressing on them we can on/off them. It will also indicate if main door is left open for more than predefined time which provides security

- **Locator Mode**

  Generally old ones have problem with memory power. So they often forget the places of their important things. So this mode is very helpful to them. This mode is basically used to find important things which is put and forgot by them. Here the device is used as transmitter and things are receivers. When the receiver catches the particular frequency transmitted by device then it will respond to it by producing the loud beeps. So they can find the things by finding

the beep direction. So there is no need to move to find the things. So this mode is very helpful to old ones. This mode is supported by both Wall mounted device and portable device so if they forgot where is the portable device they can find it by use of wall mounted device. Things are shown in screen and by just touching them alarm is activated and again touch will off it.

- **Social Networking Mode**

  When people entered into old age, their family at work makes them fill lonely. Involvement of their family and friends will help them to live happily which helps them to live longer. This mode will help them to easily contact with family and friends. They can do video calling and chatting with them. Their loved one can send mail, pictures, and videos to them. It will be very easy for them to operate mail and internet. There is also facility to store favorite songs and listen when they want to. There are some games which help them to remain mentally active. So, this mode provides connection with outer world and entertainment for them which helps to remove loneliness.

- **Emergency Mode**

  There will be one push button on the backside of the device by pressing that button will send prerecorded voice message to five of their relative set by care-taker. This will help in emergency and make possible to get treatment when most needed.

- **Setting Mode**

  This mode is very important which sets all setting of other modes.This mode is supported by only wall mounted device. In this mode there are different setting options. They can set reminders of different things like take medicines or family birthdays or doctor's appointment, Which helps them to regularly take medicine and keep record that they take medicine or not with help of 'Pill Box'? There is also record of different parameters which are monitor by different sensors like blood pressure, glucose level and weight of them.

3.2 Block Diagram of System

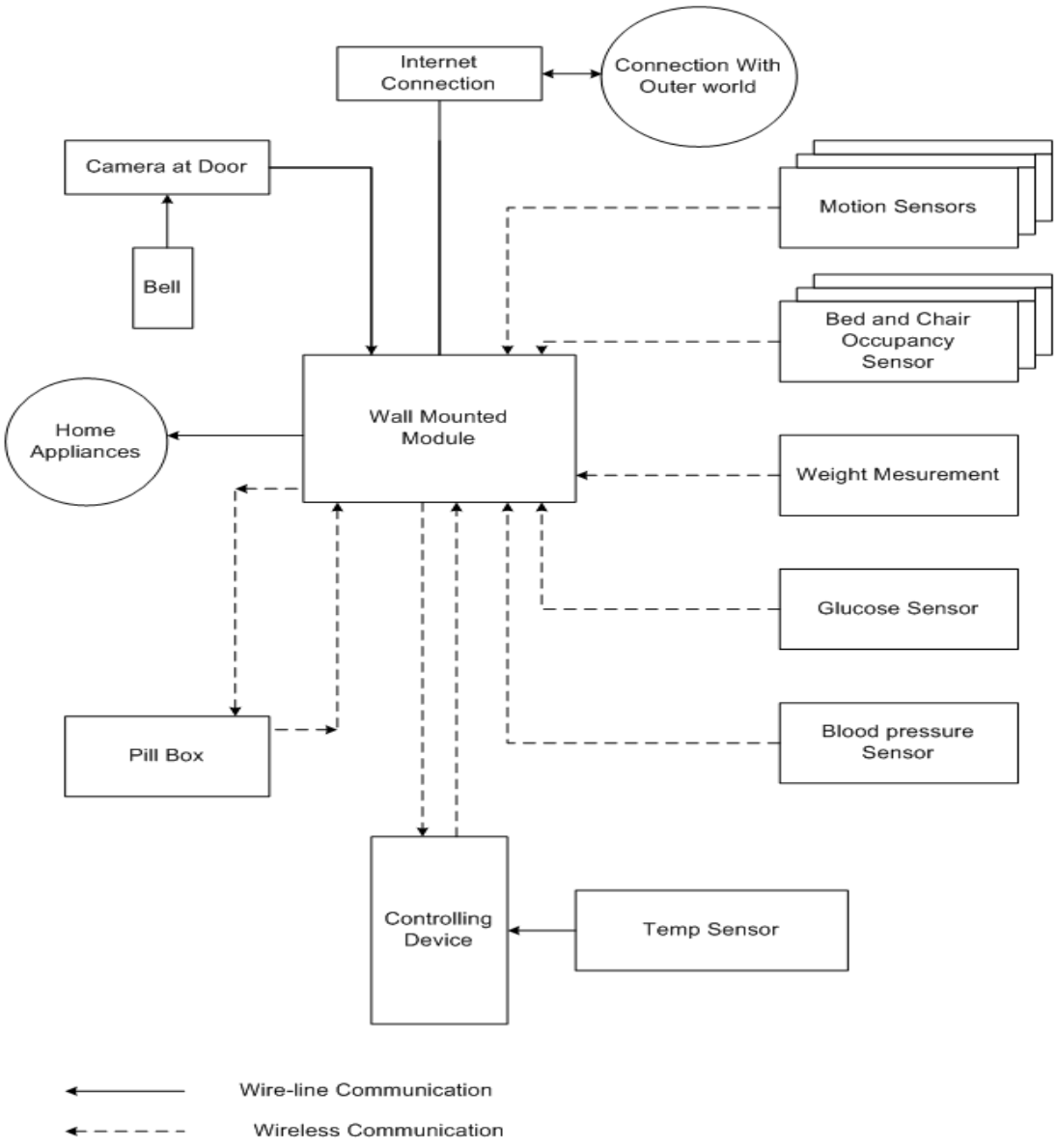

Figure 3.3: Block Diagram of System

3.3 Hardware and Software parts of Wall Mounted Device

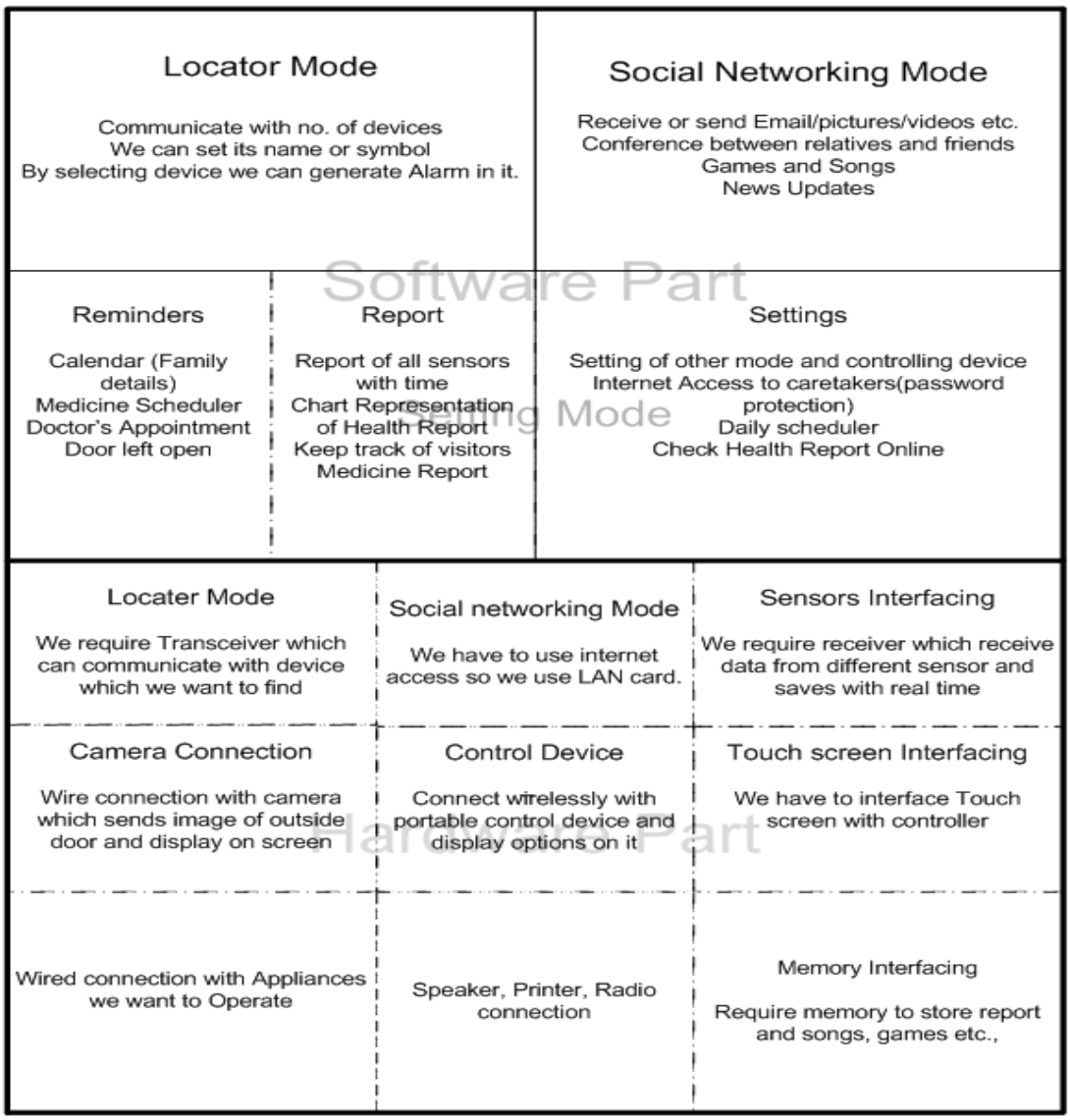

Figure 3.4: Wall Mounted Device

3.4 Hardware and Software parts of Portable Controlling Device

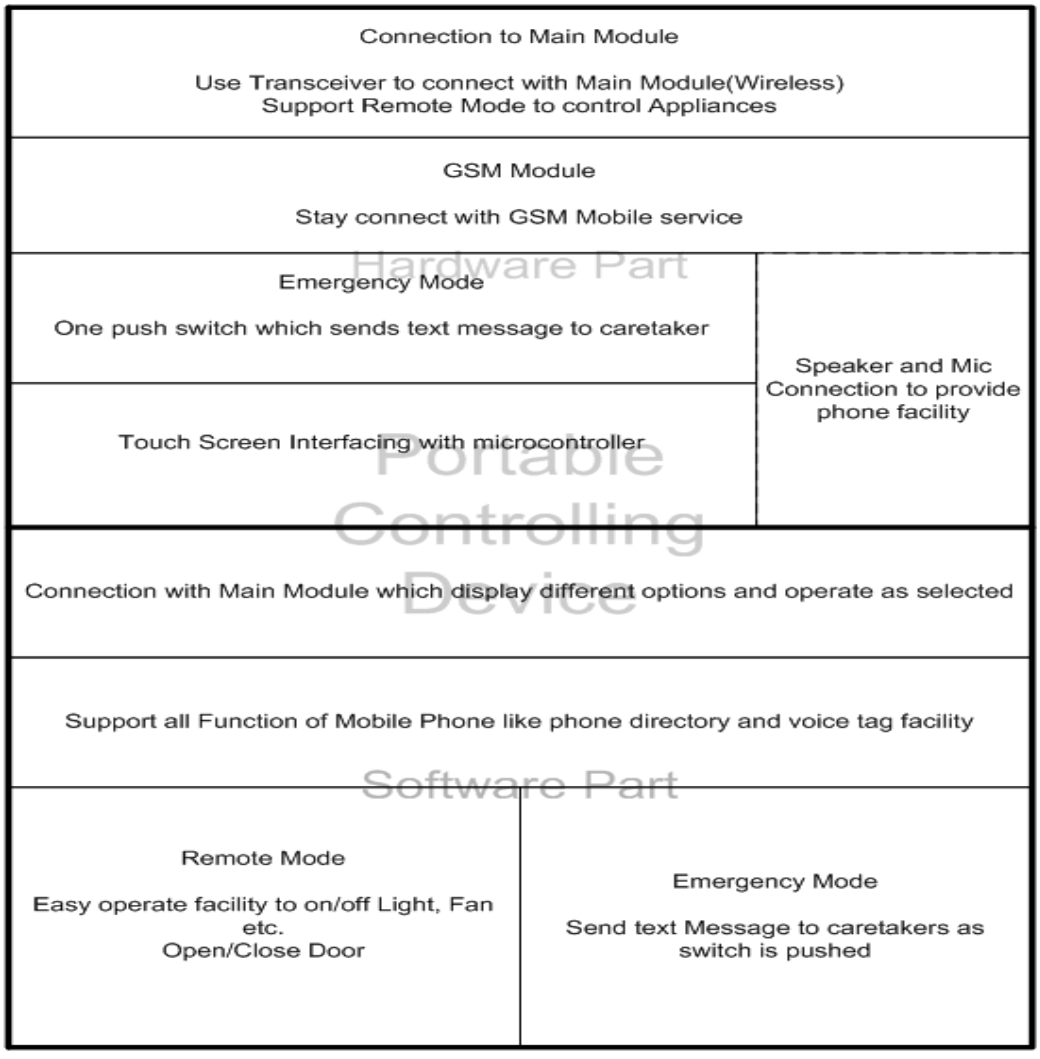

Figure 3.5: Portable Controlling Device

3.5 Summary

This chapter elaborates functionalities of Old Age Assistive Device.Different mode functionalities are also explained in detail.Seperation of hardware and software parts clearly gives an overview of how actual system will be implemented.

Chapter 4

Designing of Old Age Assistive Device

The simulation and implementation of any system requires exact work flow of the system. Designing of system is a process which is followed as part of software engineering life cycle for any system. Designing of Old Age Assistive Device is presented in this chapter. Activity diagrams are designed for each modes of operation and similarly for Wall Mounted Device and Controlling Device.Activity diagrams describe the workflow behavior of a system. Activity diagrams are similar to state diagrams because activities are the state of doing something. The diagrams describe the state of activities by showing the sequence of activities performed. Activity diagrams can show activities that are conditional or parallel.

4.1 Activity Diagrams

4.1.1 Emergency Mode

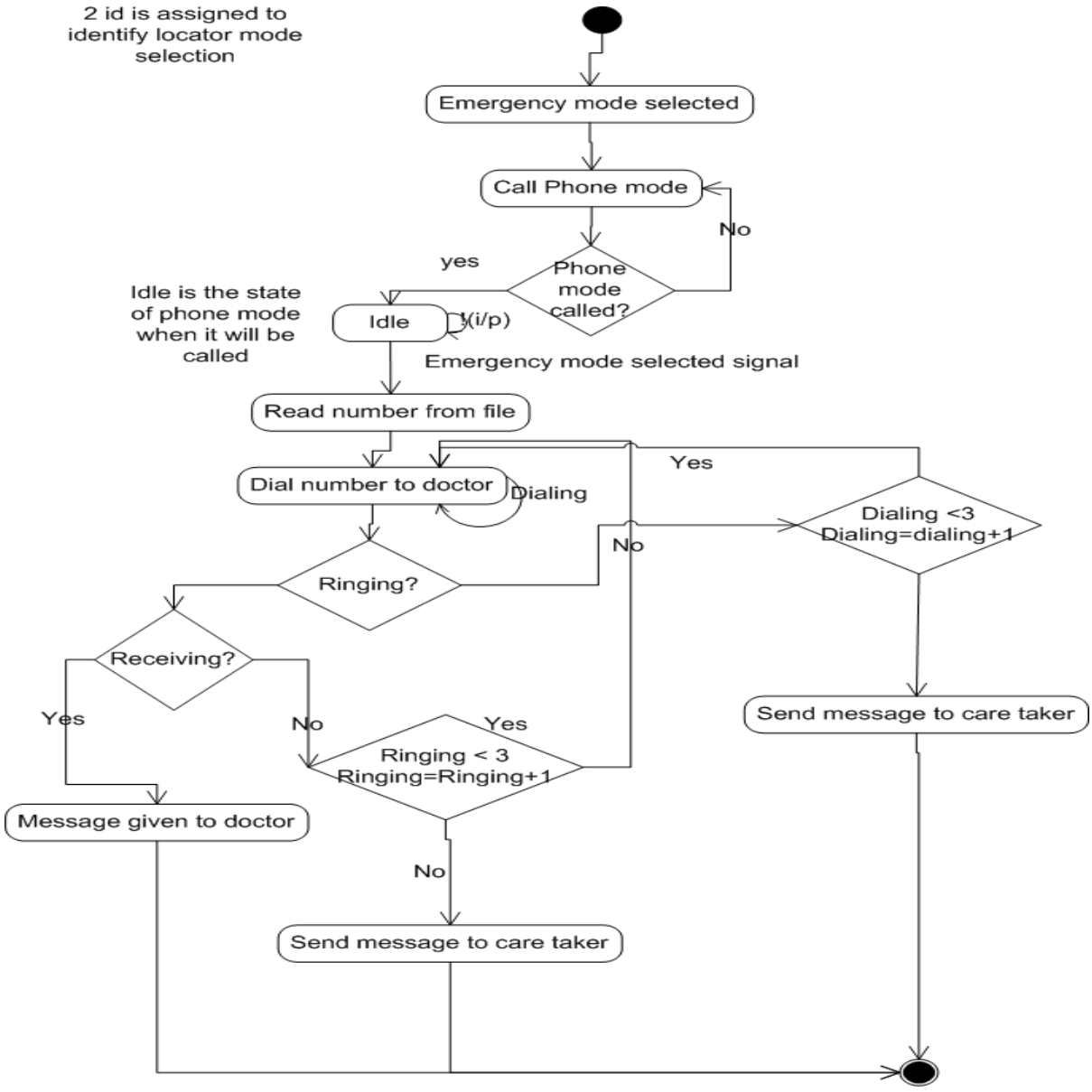

Figure 4.1: Emergency Mode

4.1.2 Phone Mode

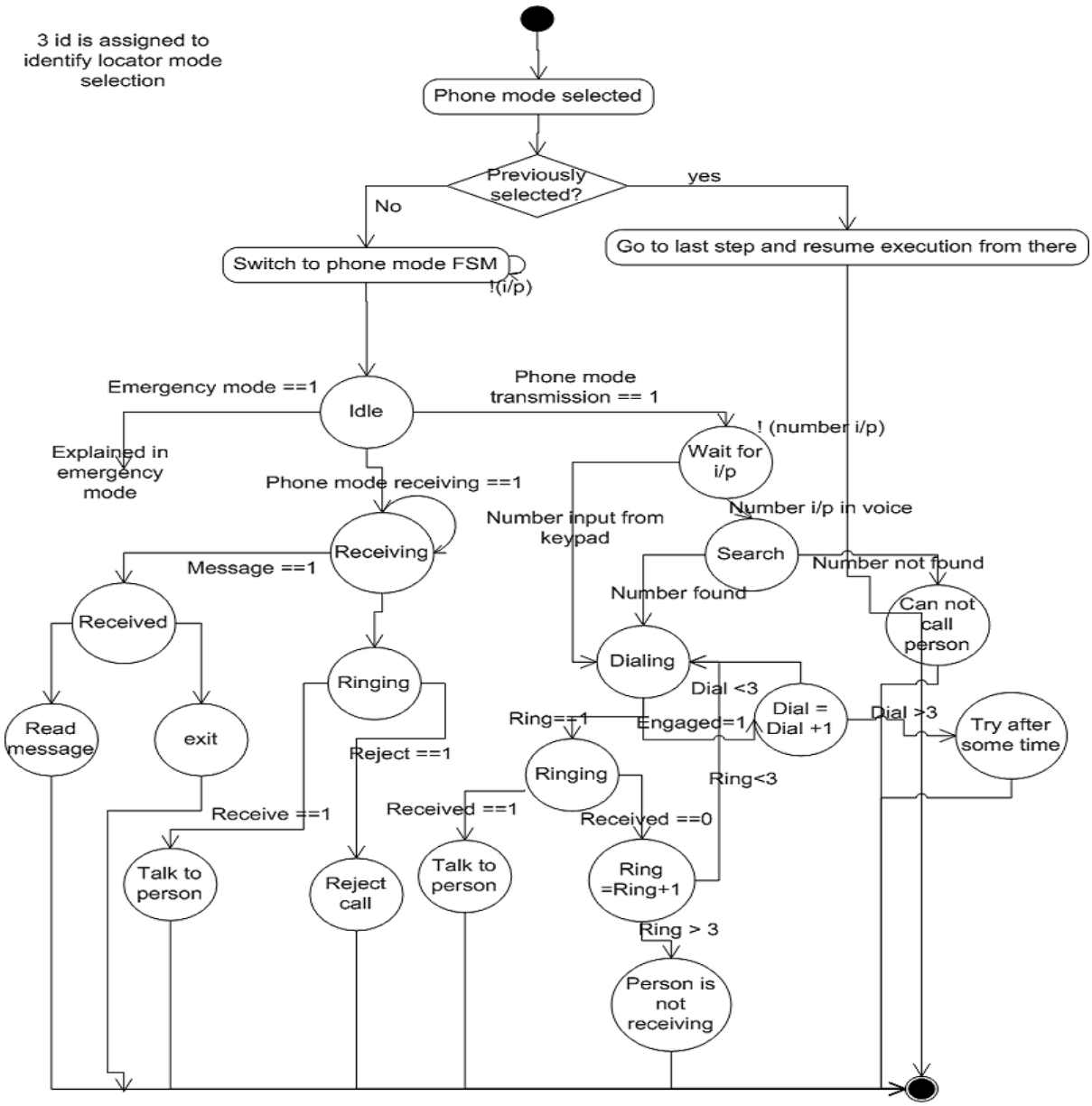

Figure 4.2: Phone Mode

4.1.3 Remote Mode

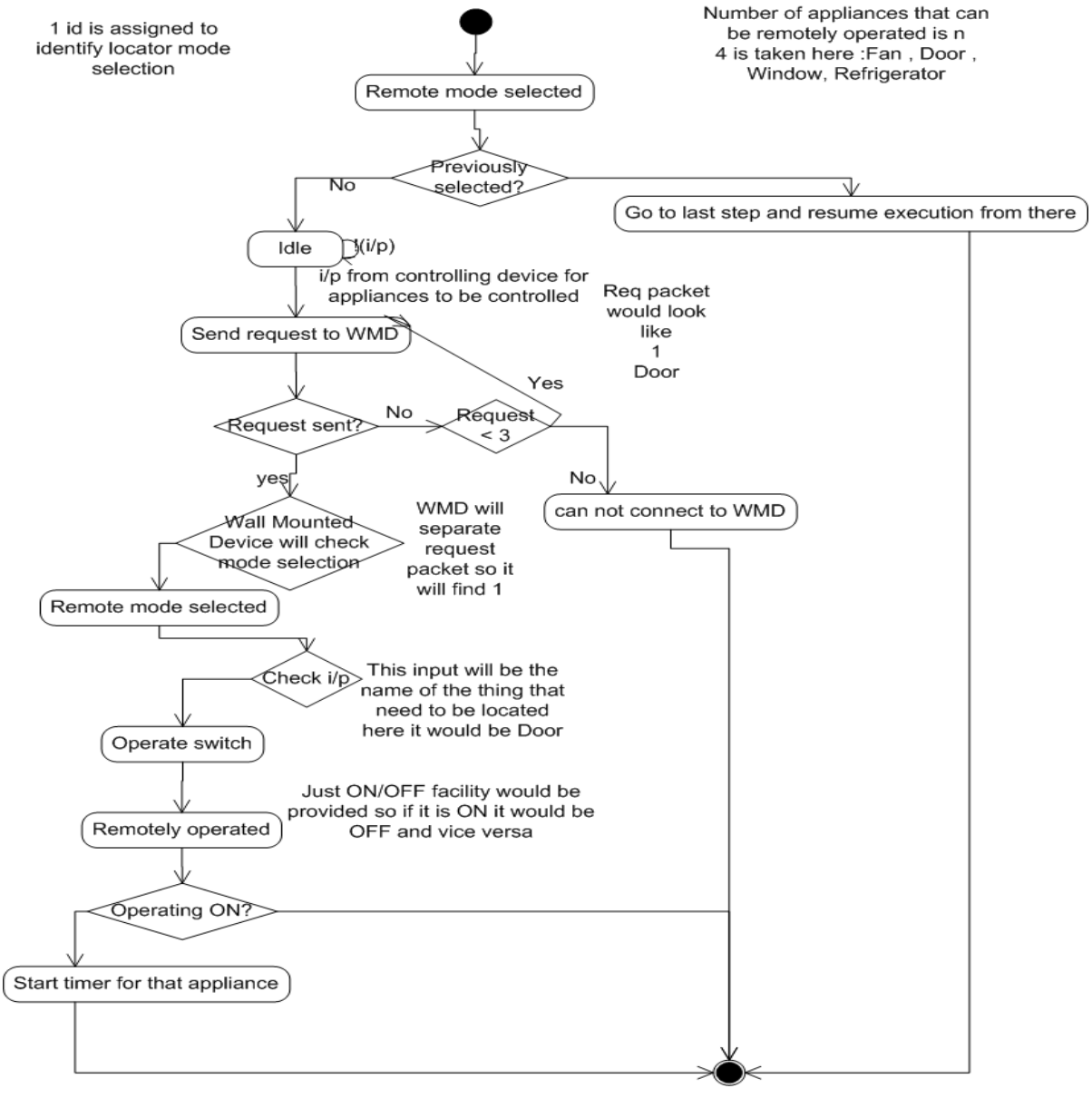

Figure 4.3: Remote Mode

4.1.4 Locator Mode

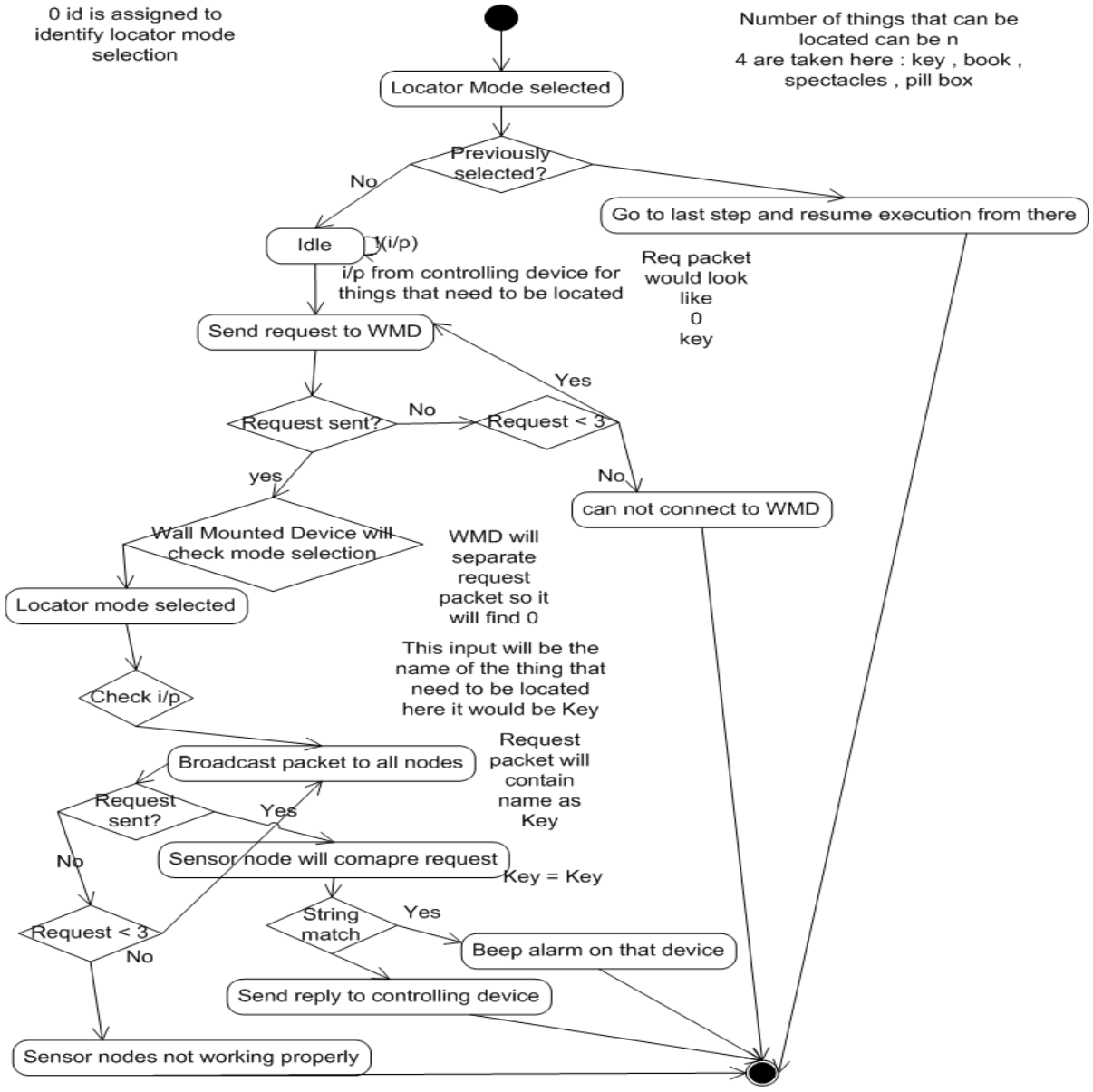

Figure 4.4: Locator Mode

4.1.5 Social Networking Mode

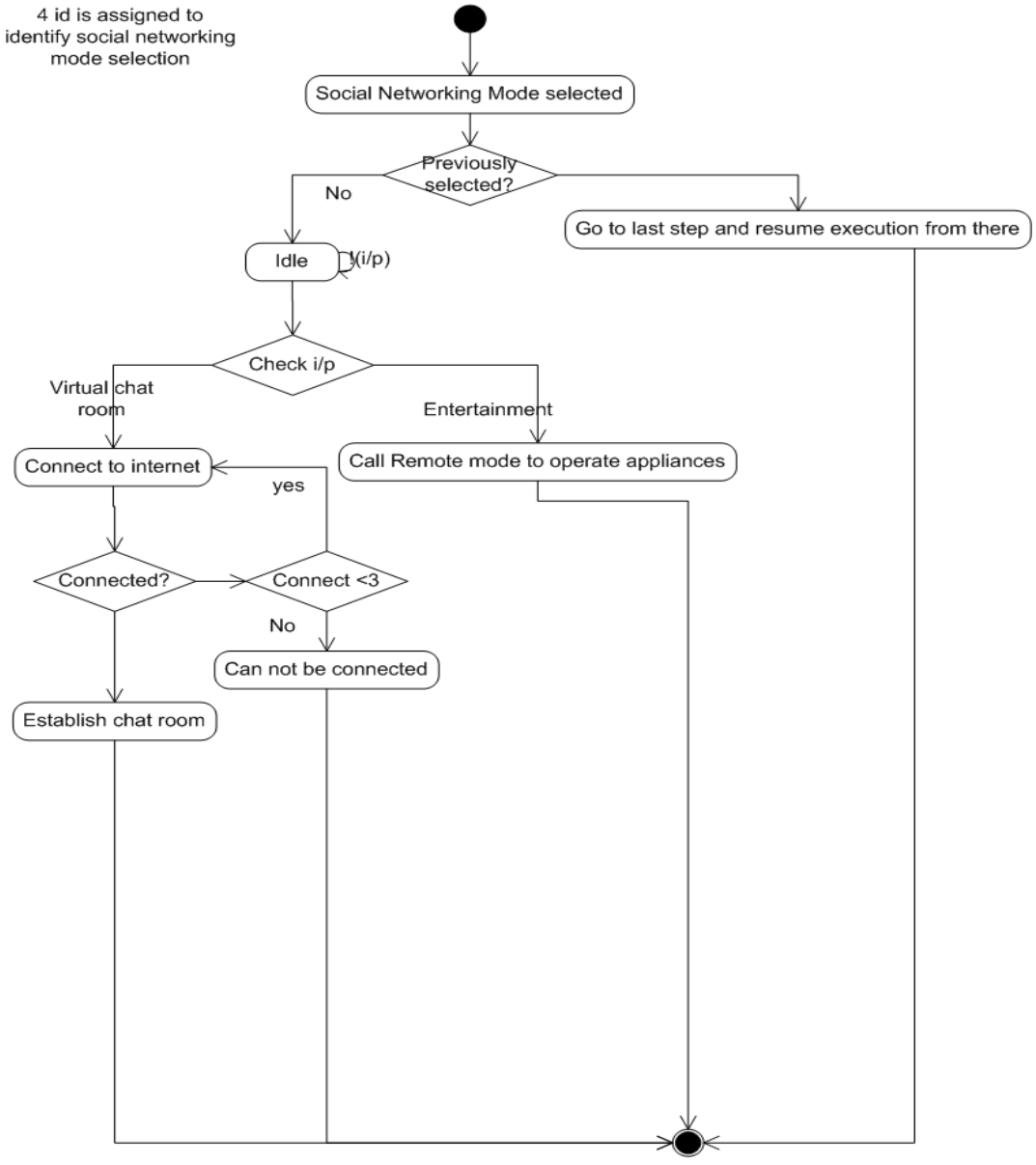

Figure 4.5: Social Networking Mode

4.1.6 Setting Mode

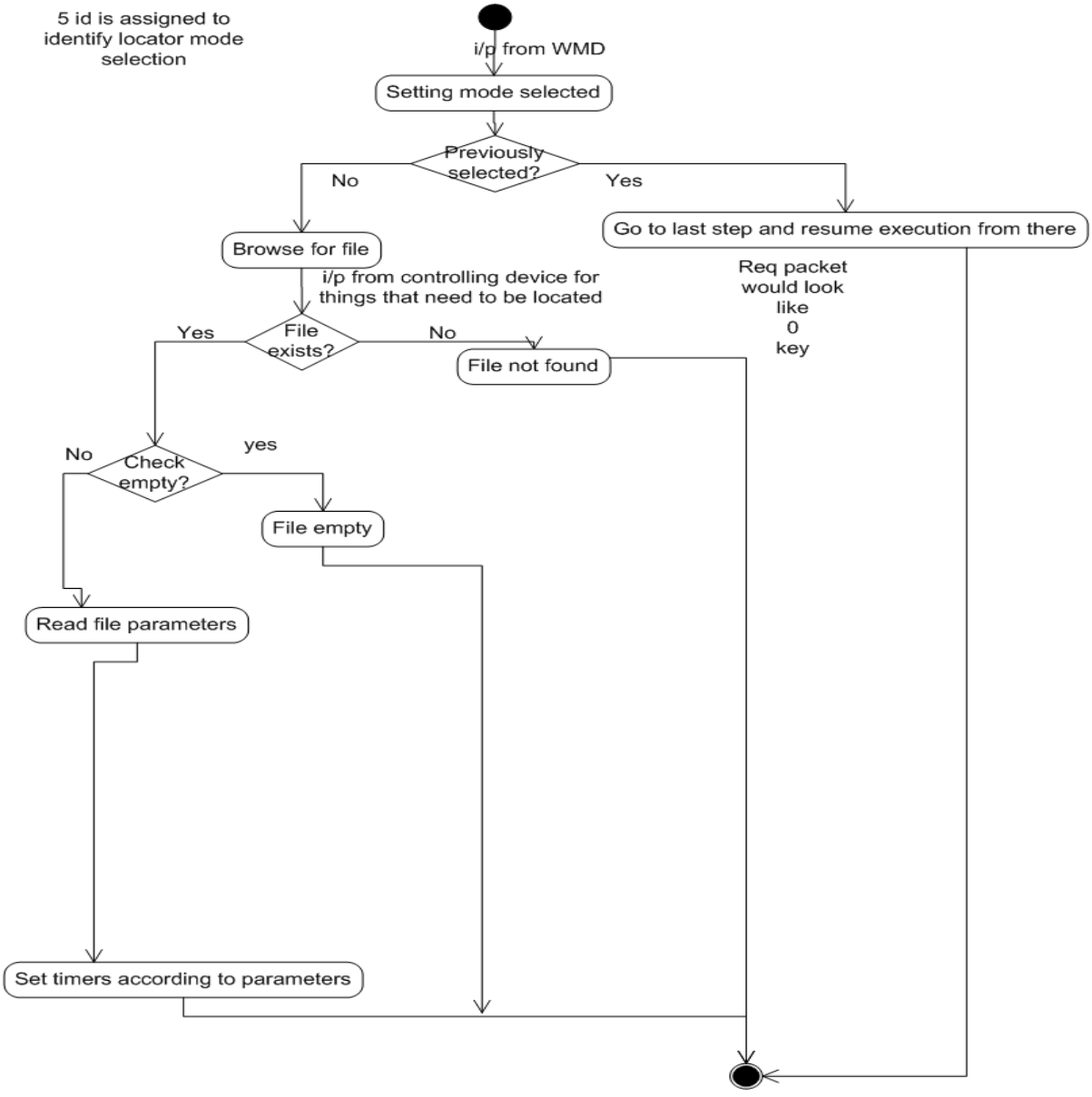

Figure 4.6: Setting Mode

4.1.7 Wall Mounted Device

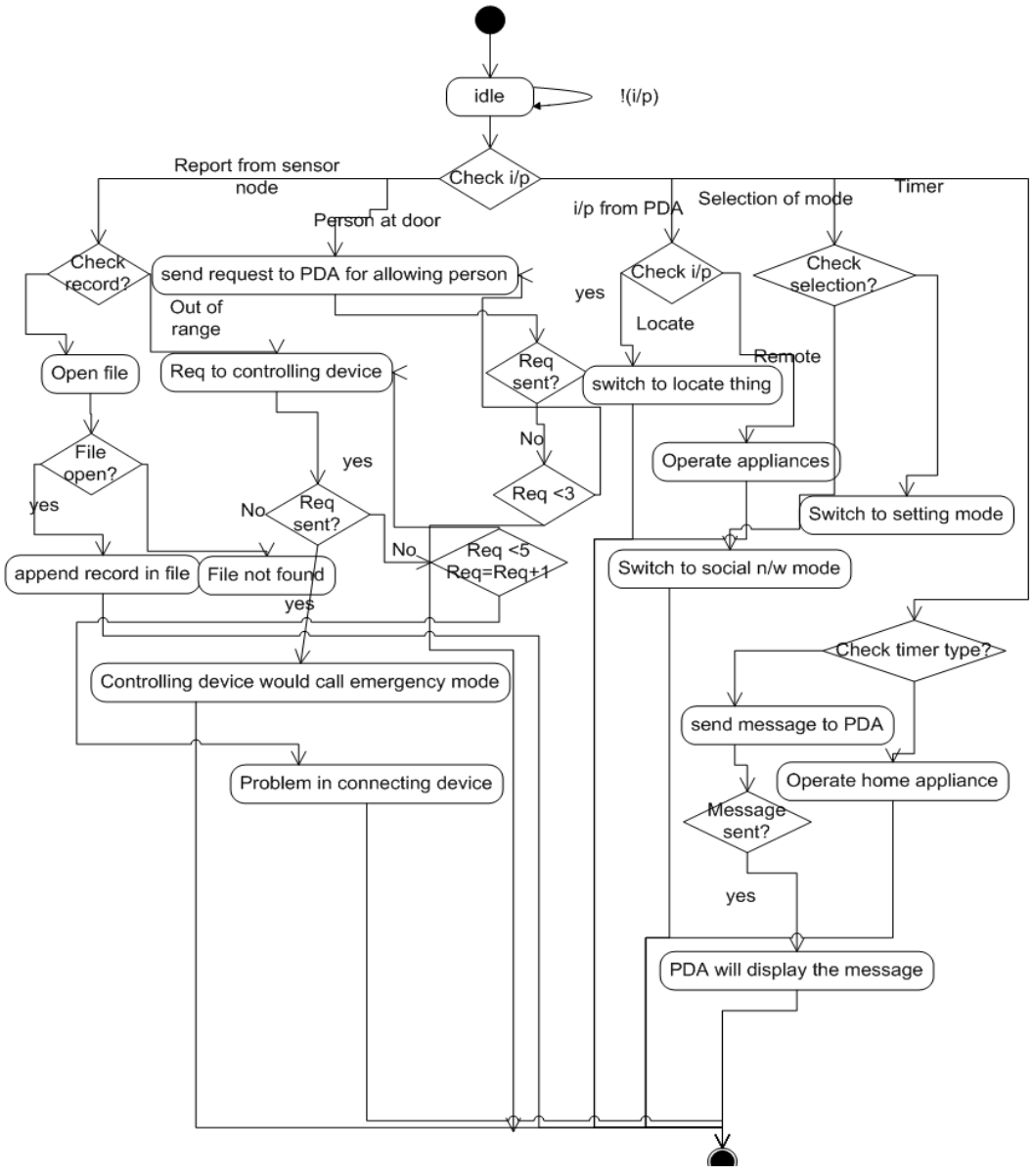

Figure 4.7: Wall Mounted Device

4.1.8 Portable Controlling Device

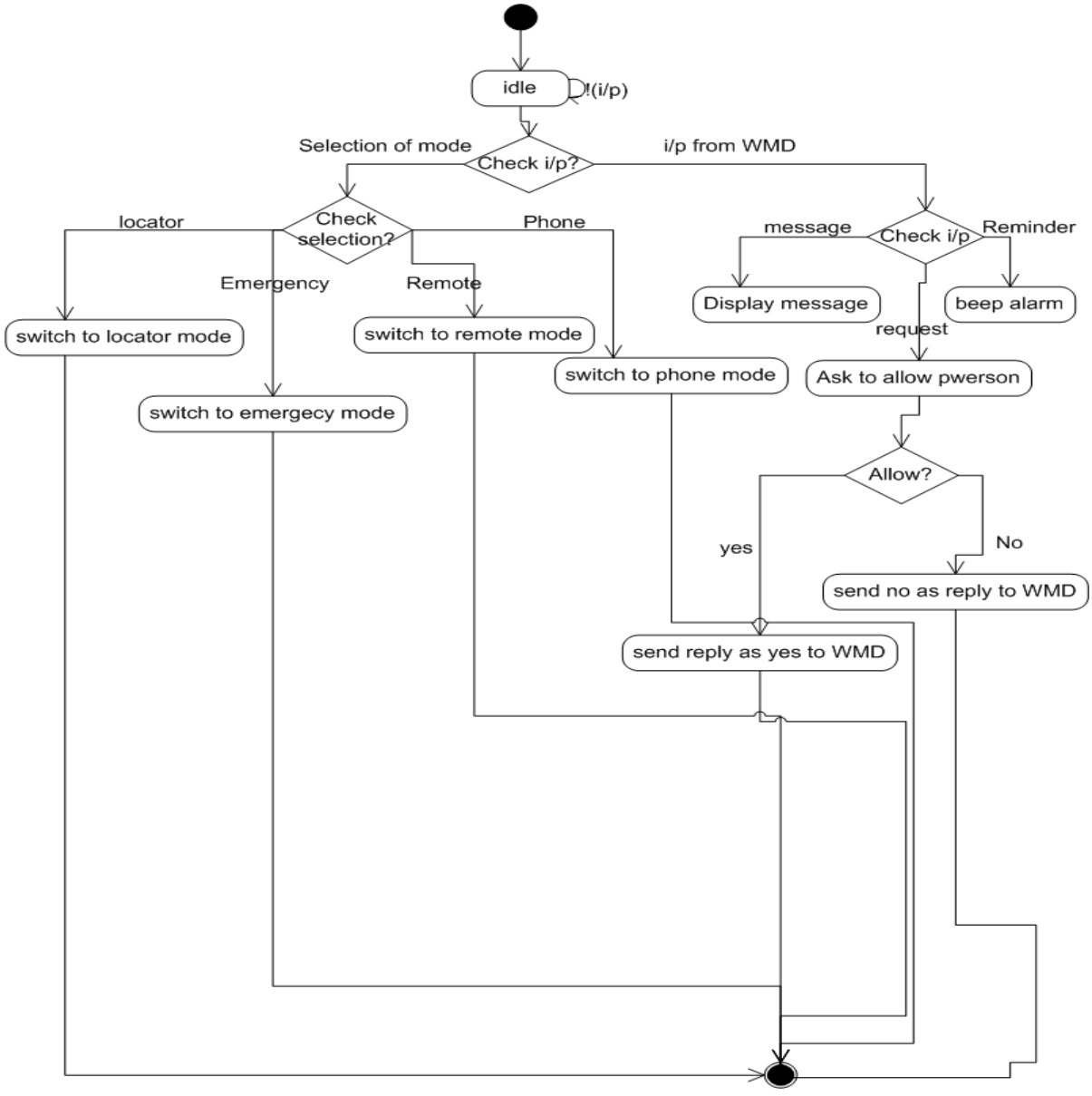

Figure 4.8: Portable Controlling Device

4.2 Summary

This chapter deals with designing Old Age Assistive Device. Simulation of Old Age Assistive Device on Ptolemy is done with respect to this diagrams.The implementation of the device would be done based on this diagrams. Different modes of operation will be called from controlling device and it will be then operated by either Wall Mounted Device or Controlling device.Some modes of operation will only be operated by Wall Mounted Device or Controlling Device.

Chapter 5

Simulation

5.1 Simulation Environment

For simulation of Old Age Assistive Device Ptolemy's visual sense tool is used which provides components for simulation wireless sensor networks and embedded systems. The study of Visual Sense is explained in Appendix A.It provides various domains which suits requirement for simulating Embedded Systems and Sensor Networks. Different modes of operation are simulated separately and then integrated with hierarchical modeling of Ptolemy to realize entire system simulation. Sensors which are require for Old Age Assistive device are simulated with primarily artificial intelligence method and added to user library of Ptolemy. Thus similar types of systems as Old Age Assistive Device can be simulated without major efforts.

For Adding required components in to Ptolemy library study of entire Ptolemy architecture was require and it is done and explained in Appendix A. Study of Viptos is done for simulation of Old Age Assistive device on Viptos which can directly generate nesC code which can be directly work on any sensor supporting tinyOS hardware.

5.2 Characteristics of Sensors

5.2.1 Temperature Sensor , Glucose sensor and Blood Pressure sensor

- Used for monitoring person's body temperature , glucose level and blood pressure level.
- It is running timer on it and sends timely signal containing value to controlling device
- Initial parameter range should be set by doctor
- Single output port will be used to send signals.
- Location of sensor will be change according to location variable

5.2.2 Motion Sensor

- Used to track persons movement in the room
- The sensor will continuously generate values of x,y coordinate and that signal will be sent to controlling device timely
- The controlling device will have plot functionality by which it will plot points on graph and it will use mathematical formula for finding distance traveled by person.
- If the person is back within time the timer will be stop else if time expires it will send signal to controlling device
- If the person is moving too faster or too slower that also can be found out by controlling device.

5.3 Simulation of Monitoring Device and Sensors

Ptolemy provides large number of library components which can be used to simulate sensors. The generic method followed for simulation of all the sensors is described.

Communication between Wall mounted device and sensors is done with wireless channel. For simulating all the sensors discrete event director of Ptolemy is used so that can be further used under wireless director of Ptolemy for system level integration.

Wall Mounted Device is used as monitoring device for all incoming sensor records

5.3.1 Steps to Simulate Monitoring Device

1. Wall Mounted device maintains different files for storing records coming from different sensors. For that it will use database manager provided by Ptolemy. It also provides component to interact with database.

2. All the sensors simulated is having clock running on it. It will send timely report to Wall Mounted Device.

3. On receiving input records timely from different sensors it will separate records and it will check that records with specified range learned with primarily artificial intelligence method and store that records in database.

4. Wall Mounted Device will send message to care taker if records are not within specified range

5. If there is some fault in sensor and if it is not working properly it will not be able to send timely record to monitoring device this will be tracked by monitoring device by running timer on it. If after certain time expires if sensor has not set any record, it will try to connect it and it will and whether it is working properly or not.

The way of simulation of monitoring device can be shown with flow chart as shown in the figure 5.1

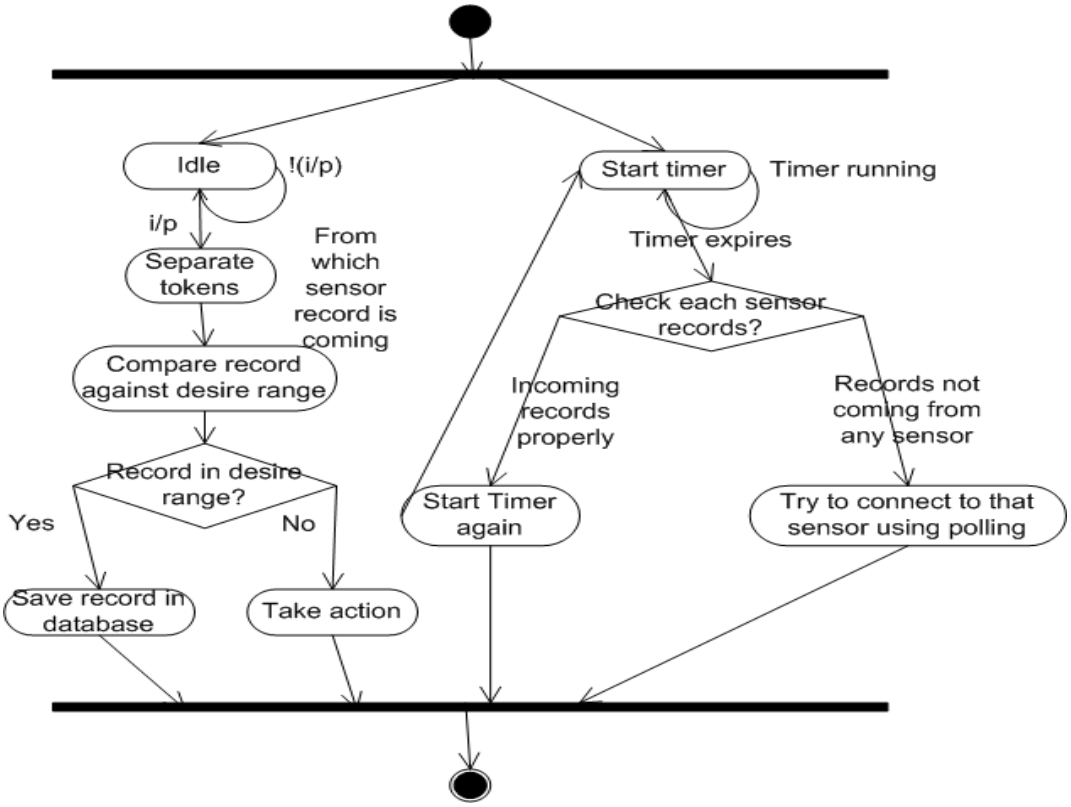

Figure 5.1: Flow chart for simulating monitoring device

5.3.2 Steps Used for Simulating Sensors using primary artificial intelligence method

1. All the sensors simulated here are working on clock. So the clock is generating an event at specified time interval. Different types of clock components are available with Ptolemy.

2. Random number generator component is used that will generate records within specified range.

3. Figure 5.2 shows flow chart of generic method used for simulation of sensors.

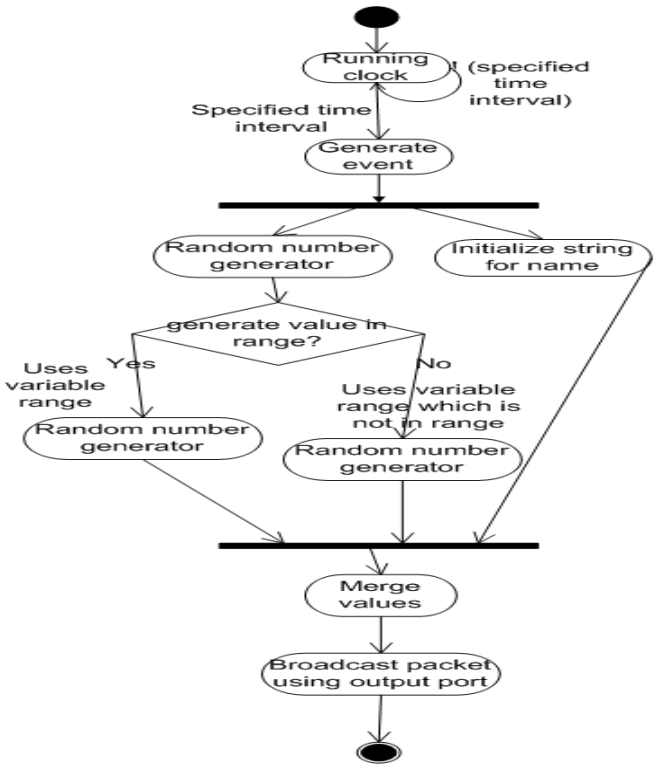

Figure 5.2: Flow chart for simulating sensors

4. Initial records are taken for learning final range variables. It will scan first ten values and then it will calculate average of those records and send that final range value to Wall Mounted Device for monitoring. This way primary artificial intelligence method is used to find the range for sensors.

5. After setting range variable by Wall mounted device , sensors will start generating values and send it to wall mounted device. The values which are generating is random so Wall Mounted Device's action can be check for correctness

6. Once sensors are simulated they are added in to library so that they can be used for simulation of similar types of systems as Old Age Assistive Device

5.4 Simulation of Modes of Operation

Below are the some details used to simulate modes of operation on Ptolemy.

1. The modes of operation of Old Age Assistive Device is simulated with use of different directors available with Ptolemy. Selection of each mode is dynamically done by generating random number. Priority of each and every node is given which will help in switching between modes when other mode will be selected while one mode is currently.

2. Hierarchical modeling is used to combine various directors which are used for as discrete event, finite state machine, wireless.

3. Emergency mode is having highest priority among all the modes of operation

4. Locator mode and Remote mode needs to operate between controlling device and Wall Mounted Device so those modes are simulated with discrete event director director under wireless director.

5. Setting mode, Social networking mode and phone mode are simulated with finite state machine. Finite state machine of phone mode is running on portable controlling device. Finite state machine for setting mode and social networking mode is running inside Wall Mounted Device

5.5 Integration of Modes using Hierarchical Modeling

1. As basic goal of Old Age Assistive Device is to communication between sensor nodes, Wireless director of Ptolemy is used as top level director of system.

2. Any sensor nodes run on discrete timing the next level in hierarchy of director is discrete event director

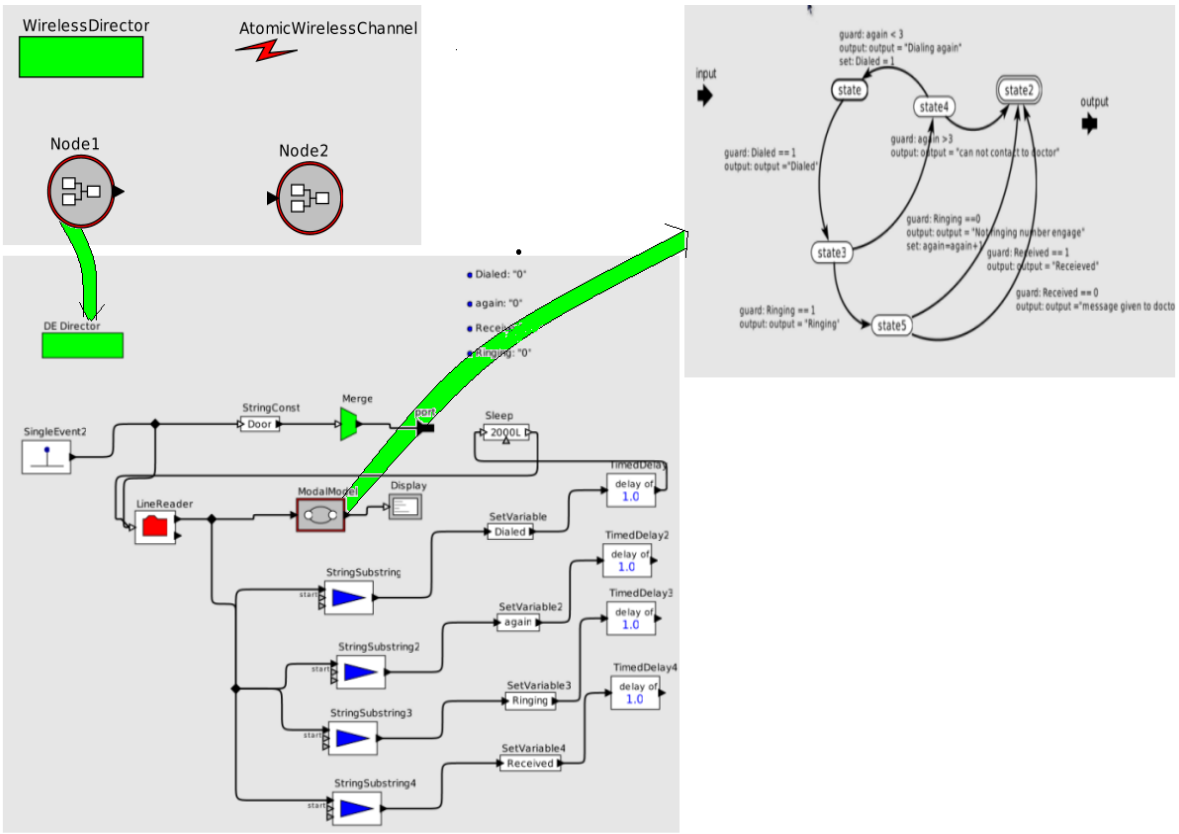

Figure 5.3: phone mode simulation using Hierarchical Modeling

3. Finite state machine is used to simulate major modes of operation so finite state machines of all modes of operation is running inside discrete event director of systems

5.6 Simulation Snapshots

1. **System level overview of Old Age Assistive Device**

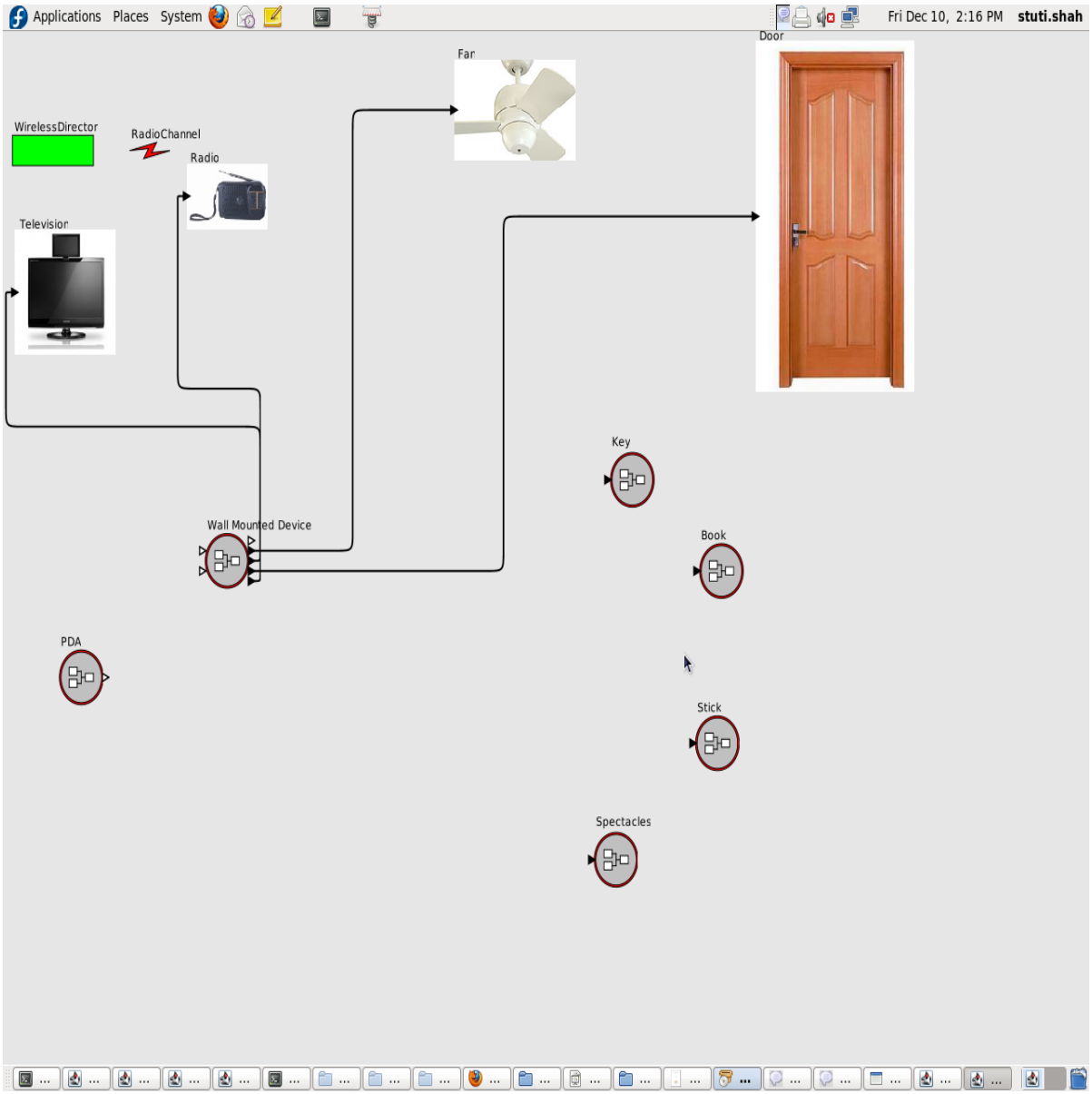

Figure 5.4: System Level Overview of Old Age Assistive Device

2. **Body Temperature Sensor**

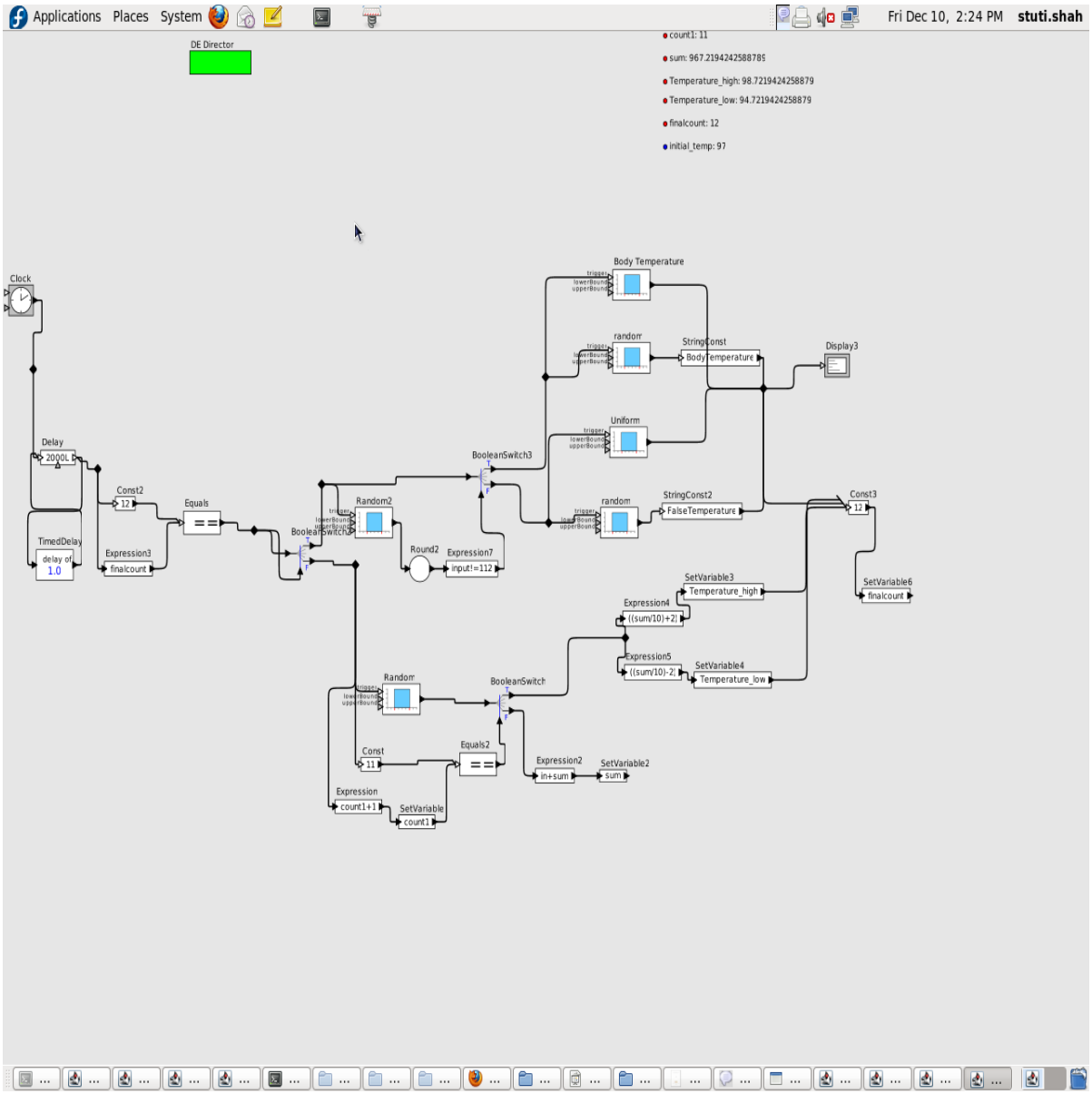

Figure 5.5: Body Temperature Sensor

3. **Phone Mode Simulated with Finite State Machine**

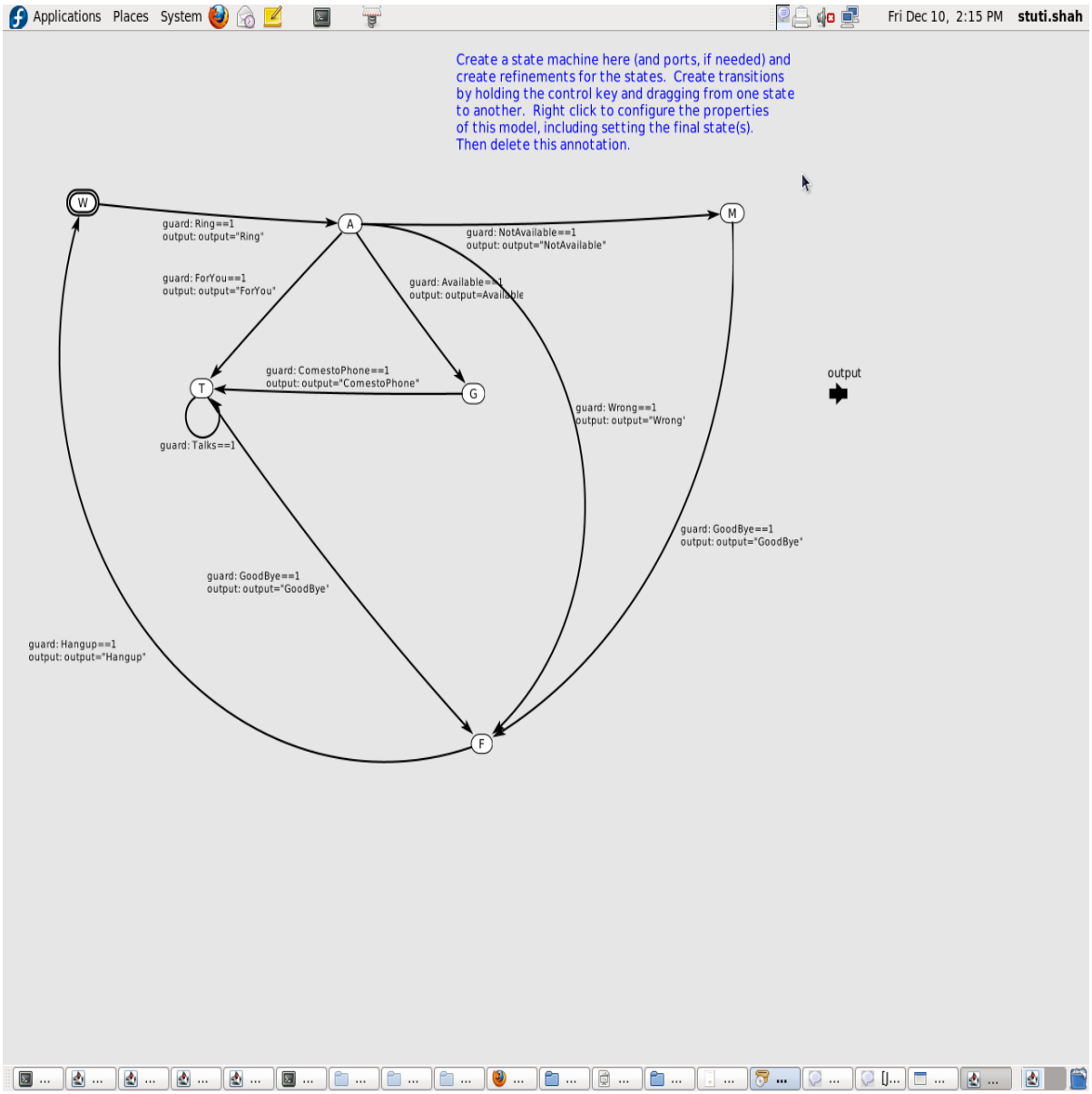

Figure 5.6: Phone Mode Simulated with Finite State Machine

4. **Sensors Added into Library**

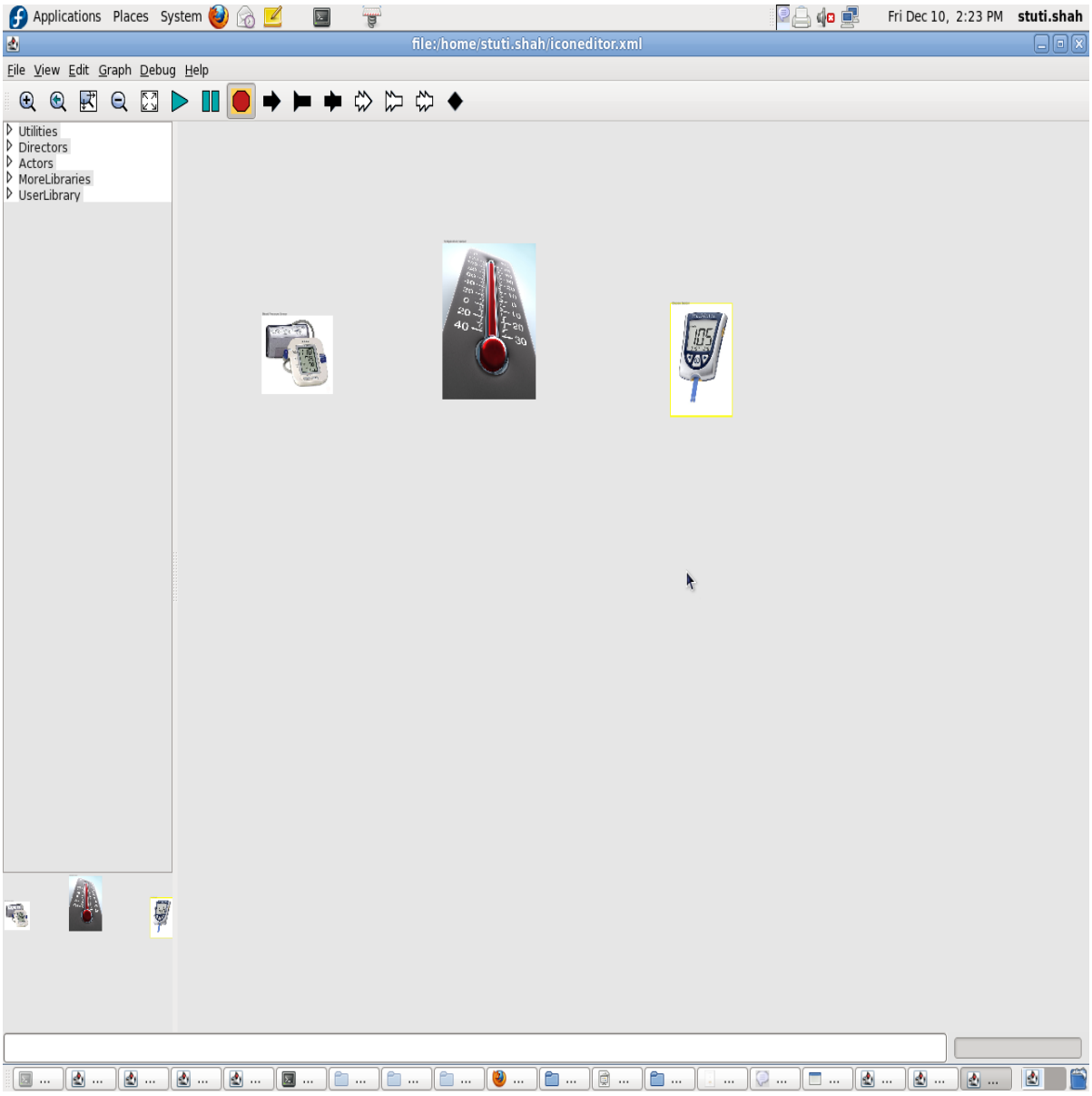

Figure 5.7: Sensors Added into Library

5.7 Simulation with Viptos

Basic goal for simulating Old Age Assistive device is to incorporate operating system for scheduling and handling interrupts on device and code generation facility available with it.

Viptos is a combination of Visual sense and TinyOS. TinyOS is event driven operating system and TinyOS hardware can run code which is written in nesC language. As TinyOS is event driven operating systems it is suitable for sensor networks and embedded systems.

Viptos contains library of components of nesC by which sensor node simulation can be done and also it provides other library of components which Visual sense is having for simulation of sensor networks. So, for simulating any system the actual sensor node code can be realized and can be checked for correctness with rest part of the system which is simulated with visual sense.

5.8 Summary

This chapter summarize different approach used for simulating Old Age Assistive Device on Ptolemy.The architecture study of Ptolemy is done for adding components in to Ptolemy library. Old Age Assistive device is simulated with hierarchial modeling approach. Viptos will be used for simulation of sensor nodes and generating nesC code.

Chapter 6

Implementation

This chapter deals with implementation of Old Age Assistive Device on hardware. PSOC (Programmable System on Chip) developer board is used for hardware implementation. Main goal is to used model based approach for implementation

6.1 Finite State Machine on PSOC

Any system can be realized with finite state machine. modes of operation of Old Age Assistive Device are simulated with finite state machine on Ptolemy.PSOC IDE (Integrated Development Environment) gives flexibility to input finite state machine as transfer function between input and output drivers. Other available transfer functions are priority encoder, status encoder,table lookup and set point region. Below are the steps to implement Locator Mode of Old Age Assistive Device on PSOC

1. Number of things that can be located so that many number of buttons are require. so four buttons from input driver is taken

2. when button will be pressed the transfer function will work and it will place output on output driver

3. For Locator mode output driver will be wireless USB, in which data will change according to transfer function FSM.

4. Every thing which can be located is given identity number so when that button is pressed accordingly number will be placed on wireless.

5. Other Finite State Machine will be running on receiver's end which will identify data placed in wireless and beep alarm if that data is for the receiving node itself.

Figure 6.1shows the implementation of locator mode on sender side

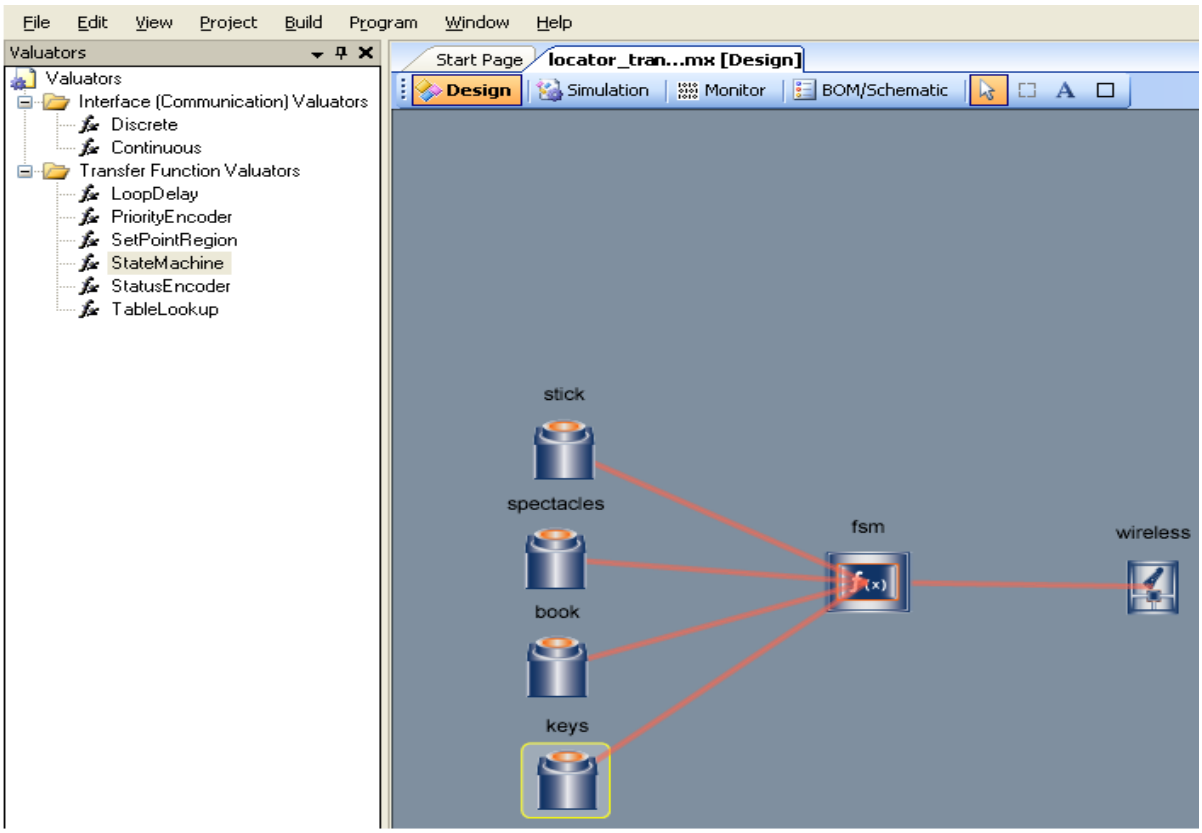

Figure 6.1: Sender implementation of Locator Mode

Figure 6.2 shows finite state machine running on sender side.

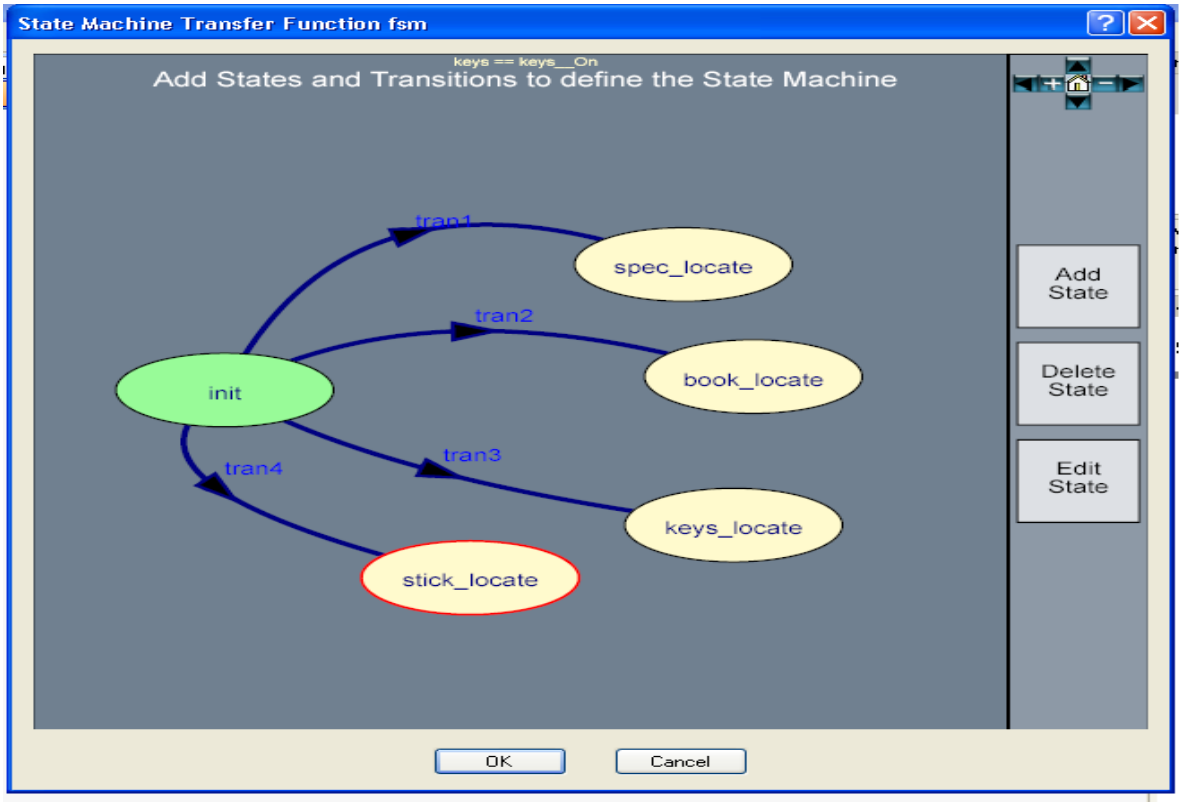

Figure 6.2: Finite State Machine on Sender Side

PSOC evaluation board has four different processor which can load different programs and can run simultaneous programs once loaded. For wireless connectivity radio connectors are given. So by interfacing two radio to two different processor of the same board one can work as controlling device and other can work as receiver end. PSOC also gives flexibility for communication between different processors so one processor can work as master and other can work as slave. I2C bus is used for communication between master and slave.

Figure 6.3 shows Receiver end implementation

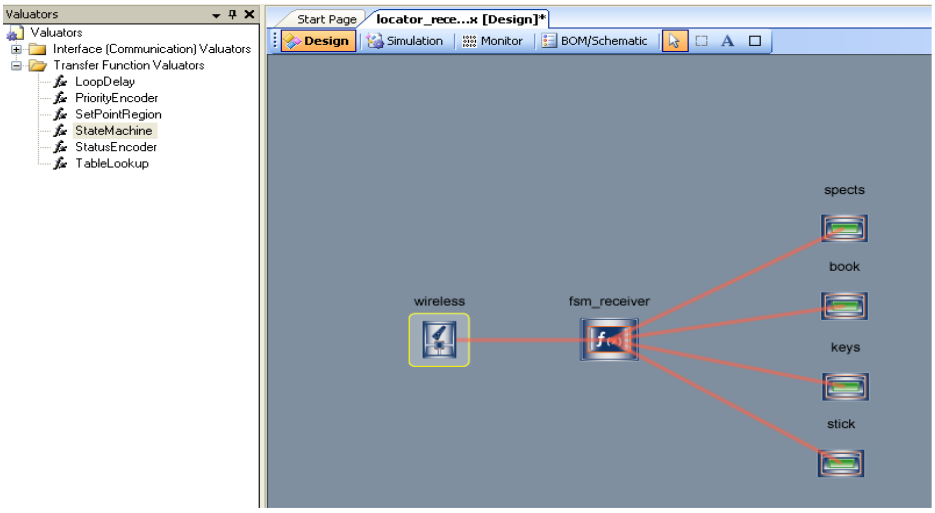

Figure 6.3: Receiver implementation of Locator Mode

Receiver finite state machine implementation is show in figure 6.4

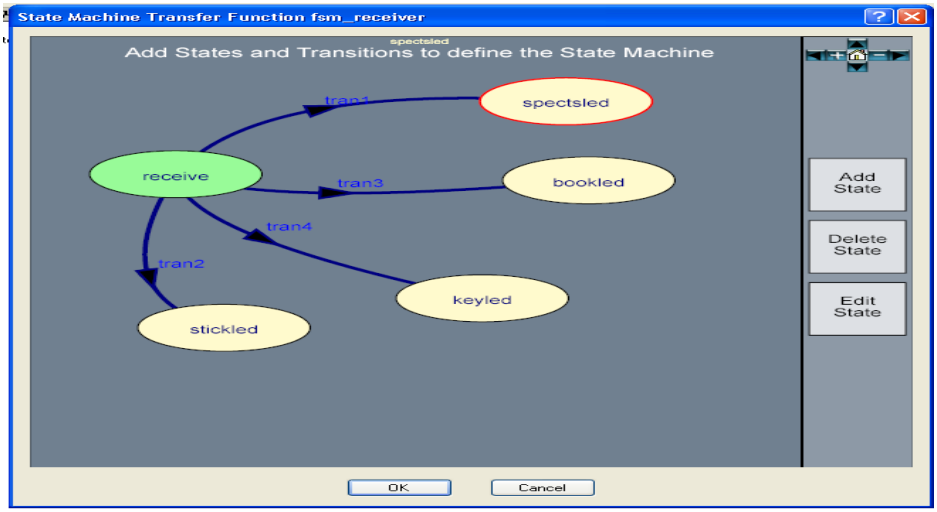

Figure 6.4: Finite State Machine at Receiver end

6.2 Summary

This chapter summarize the approach for implementing Old Age Assistive Device on hardware. Finite State Machines which are implemented as part of Ptolemy model are implemented on PSOC developer board.All modes of operation of Old Age Assistive Device can be realized with the help of finite state machine.Ptolemy finite state machine also provides code generation facility which can generate c code which can be use for implementation on hardware.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis focuses on mainly two objectives one is designing of modeling tool for Embedded Systems and Sensor Networks and to implement Old Age Assistive device as case study.In literature survey various open source modeling tools are studied. The requirement for designing embedded systems and sensor networks is found out and this features are compared with features of various modeling tools available. From various modeling tools studied so far I concluded Ptolemy as best available modeling tool available for simulating Embedded Systems and Sensor Networks.Modes of operation of Old Age Assistive device is implemented on Ptolemy using various directors and tools such as VisualSense.Hiearchical modeling of Ptolemy is used to combine various directors and to realize entire system.The sensors which are require for Old Age Assistive Device are implemented with primary artificial intelligence method and added to Ptolemy library.Thus similar types of systems such as Old Age Assistive Device can be simulated with out major efforts. Code generation facility of Ptolemy is used for achieving second objective which is implement Old Age Assistive device on hardware. Finite state machine which are implemented as part of Ptolemy model is used for implementing Old Age Assistive Device on hardware. Thus reducing time

require to design and implement any system.

7.2 Future Scope

The future scope for this project can be enhancing Ptolemy which can give code generation facility in for more domains which is not available. Using Viptos for implementing system can avoid requirement for writing nesC code for any sensor node so Integrating Ptolemy and Viptos model can be one direction of working by which model based approach can be fully satisfied. Adding modules to Old Age Assistive Device as per needs of society can be future work for Old Age Assistive Device

Chapter 8

List of publication

1. Published "Implementation of Sensors using Ptolemy in International Journal of Computer Science and Emerging Technologies (IJCSET),Volume 2,Issue 2,April-2011,ISSN 2044-6004

2. Presented "Survey of Modeling Tools" at International Conference on Systemics,Cybernetics and Informatics(ICSCI)2011,Taramati Baradari Cultural Complex, Ibrahimbagh, hyderabad

3. Accepted paper Titled "Implementation of Hierarchical Model in Ptolemy" in WIMON 2011 conference (To be published in Springer CCIS proceedings), Park Hotels, Chennai.

4. Accepted paper Titled "Health Monitoring Sensors implementation using Ptolemy" in NeCoM 2011,(To be published in Springer CCIS proceedings), Park Hotels, Chennai.

Appendix A

Ptolemy, VisualSense, Viptos

A.1 Ptolemy Architecture

Ptolemy II offers an infrastructure for implementations of a number of models of computation. The overall architecture consists of a set of packages that provide generic support for all models of computation and a set of packages that provide more specialized support for particular models of computation. Examples of the former include packages that contain math libraries, graph algorithms, an interpreted expression language, signal plotters, and interfaces to media capabilities such as audio. Examples of the latter include packages that support clustered graph representations of models, packages that support executable models, and domains, which are packages that implement a particular model of computation.

Ptolemy II is modular, with a careful package structure that supports a layered approach. The core packages support the data model, or abstract syntax, of Ptolemy II designs. They also provide the abstract semantics that allows domains to inter operate with maximum information hiding. The UI packages provide support for our XML file format, called MoML, and a visual interface for constructing models graphically. The library packages provide actor libraries that are domain polymorphic, meaning that they can operate in a variety of domains. And finally, the domain

packages provide domains, each of which implements a model of computation, and some of which provide their own, domain-specific actor libraries.

A.1.1 Core Packages

The role of each package is explained below. **actor** : This package supports executable entities that receive and send data through ports. It includes both untyped and typed actors. For typed actors, it implements a sophisticated type system that supports polymorphism. It includes the base class Director that is extended in domains to control the execution of a model.

actor.lib : This subpackage and its subpackages contain domain polymorphic actors.

actor.parameters : This subpackage provides specialized parameters for specifying locations, ranges of values, etc.

actor.process : This subpackage provides infrastructure for domains where actors are processes implemented on top of Java threads.

actor.sched : This subpackage provides infrastructure for domains where actors are statically scheduled by the director, or where there is static analysis of the topology of a model associated with scheduling.

actor.util : This subpackage contains utilities that support directors in various domains. Specifically, it contains a simple FIFO Queue and a sophisticated priority queue called a calendar queue.

copernicus : This subpackage contains the actor specialization infrastructure (Java code generation)

data : This package provides classes that encapsulate and manipulate data that is transported between actors in Ptolemy models. The key class is the Token class, which defines a set of polymorphic methods for operating on tokens, such as add(), subtract(), etc.

data.expr : This class supports an extensible expression language and an in-

terpreter for that language. Parameters can have values specified by expressions. These expressions may refer to other parameters. Dependencies between parameters are handled transparently, as in a spreadsheet, where updating the value of one will result in the update of all those that depend on it.

data.type : This package contains classes and interfaces for the type system

graph : This package and its subpackage, graph.analysis, provides algorithms for manipulating and analyzing mathematical graphs. This package is expected to supply a growing library of algorithms. These algorithms support scheduling and analysis of Ptolemy II models.

kernel : This package provides the software architecture for the Ptolemy II data model, or abstract syntax. This abstract syntax has the structure of clustered graphs. The classes in this package support entities with ports, and relations that connect the ports. Clustering is where a collection of entities is encapsulated in a single composite entity, and a subset of the ports of the inside entities are exposed as ports of the composite entity.

kernel.attributes : This subpackage of the kernel package provides specialized attributes such as File- Attribute, which is used in actors to specify a file or URL.

kernel.undo : This subpackage of the kernel package provides facilities for associating with a model a record of actions performed on it and their undo mechanisms.
kernel.util : This subpackage of the kernel package provides a collection of utility classes that do not depend on the kernel package. It is separated into a subpackage so that these utility classes can be used without the kernel. The utilities include a collection of exceptions, classes supporting named objects with attributes, lists of named objects, a specialized cross-reference list class, and a thread class that helps Ptolemy keep track of executing threads.

math : This package encapsulates mathematical functions and methods for operating on matrices and vectors. It also includes a complex number class, a class supporting fractions, and a set of classes supporting fixed-point numbers.

matlab : This package contains the MATLAB interface.

util : This package contains various Ptolemy-independent utilities, such as string utilities and XML utilities.

A.1.2 Overview of Key Classes

Instances of all of the classes can have names; they all implement the Nameable interface. Most of the classes generalize NamedObj, which in addition to being nameable can have a list of attributes associated with it. Attributes themselves are instances of NamedObj.

Entity, Port, and Relation are three key classes that extend NamedObj, directly or indirectly. These classes define the primitives of the abstract syntax supported by Ptolemy II.ComponentPort, ComponentRelation, and ComponentEntity extend these classes by adding support for clustered graphs. CompositeEntity extends ComponentEntity and represents an aggregation of instances of ComponentEntity and ComponentRelation.

The actor-oriented class mechanism is represented by the Derivable and Instantiable interfaces. Nearly all classes implement Derivable, which means that they can be contained in class definitions and their presence in subclasses and instances will be implied by their presence in the class definition. The InstantiableNamedObj subclass of NamedObj is the base class for objects that can serve as class definitions, subclass definitions, or instances of classes. Currently, only Entity and its subclasses can play the role of a class or instance.

The Executable interface defines objects that can be executed. The Actor interface extends this with capability for transporting data through ports. AtomicActor and CompositeActor are concrete classes that implement this interface. The Executable and Actor interfaces are key to the Ptolemy II abstract semantics.

An executable Ptolemy II model consists of a top-level CompositeActor with an instance of Director and an instance of Manager associated with it. The manager provides overall control of the execution (starting, stopping, pausing). The director

implements a semantics of a model of computation to govern the execution of actors contained by the CompositeActor.

Director is the base class for directors that implement models of computation. Each such director is associated with a domain. As explained above, we have defined in Ptolemy II directors that implement continuous-time modeling (ODE solvers), process networks, synchronous dataflow, discrete event modeling, and communicating sequential processes

A.1.3 Domains

The domains in Ptolemy II are subpackages of the ptolemy.domains package. These packages generally contain a kernel subpackage, which defines classes that extend those in the actor or kernel packages of Ptolemy II. The lib subpackage, when it exists, includes domain-specific actors.

A.1.4 Library Packages

Most domains extend classes in the actor package to give a specific semantic interpretation to an interconnection of actors. It is possible, and strongly encouraged, to define actors in such a way that they can operate in multiple domains. Such actors are said to be domain polymorphic. These packages are briefly described below:

actor.corba: This package includes actors and infrastructure for distributed models that use CORBA.

actor.lib : This package is the main library of polymorphic actors.

actor.lib.comm : This is a library of actors for modeling communications systems.

actor.lib.conversions : This package provides domain polymorphic actors that convert data between different types.

actor.lib.gui : This package is a library of polymorphic actors with user interface components, such as plotters.

actor.lib.hoc : This package is a library of higher-order components, which are components that construct portions of a model.

actor.lib.image: This package is a library of image processing actors that does not depend on JAI or JMF.

actor.lib.io : This package provides file I/O.

actor.lib.io.comm : This package provides an actor that communicate via the serial ports. This actor works only under Windows.

actor.lib.jai : This is a library of image processing actors based on the Java advanced imaging (JAI)API .

actor.lib.javasound : This package provides sound actors.

actor.lib.jmf : This is a library of image processing actors based on the Java media framework API.

actor.lib.joystick : This package provides an example actor that communicates with a particular I/O device, a joystick.

actor.lib.jxta : This is a library of experimental actors supporting the JXTA discovery mechanism from Sun Microsystems.

actor.lib.logic : This package provides actors that perform logical functions like AND, OR and NOT.

actor.lib.net : This package provides actors that communicate using datagrams.

actor.lib.python : This package provides an actor whose operation can be specified in Python.

A.1.5 User Interface Packages

The UI packages provide support for our XML file format, called MoML, and a visual interface for constructing models graphically, called Vergil. The intent of each package is described below:

actor.gui : This package contains the configuration infrastructure, which supports modular construction of user interfaces that are themselves Ptolemy II models.

actor.gui.style : This package contains classes that decorate attributes to serve as hints to a user interface about how to present these attributes to the user.

gui : This package contains generically useful user interface components.

media : This package encapsulates a set of classes supporting audio and image processing.

media.javasound : This package encapsulates a set of classes supporting audio and processing that depends on the JavaSound API

moml : This package contains classes support our XML modeling markup language (MoML), which is used to describe Ptolemy II models.

moml.filter : This package provides backward compatibility between Ptolemy release. plot This package and its subpackages provide two-dimensional signal plotting widgets. vergil This package and its subpackages contain the Ptolemy II graphical user interface. It builds on Diva, a toolkit that extends Java 2D.

A.2 VisualSense

VisualSense is a modeling and simulation framework for wireless and sensor networks that builds on and leverages Ptolemy II. Modeling of wireless networks requires sophisticated representation and analysis of communication channels, sensors, ad-hoc networking protocols, localization strategies, media access control protocols, energy consumption in sensor nodes, etc. This modeling framework is designed to support a component-based construction of such models. It supports actor-oriented definition of network nodes, wireless communication channels, physical media such as acoustic channels, and wired subsystems. The software architecture consists of a set of base classes for defining channels and sensor nodes, a library of subclasses that provide certain specific channel models and node models, and an extensible visualization framework. Custom nodes can be defined by subclassing the base classes and defining the behavior in Java or by creating composite models using any of several Ptolemy II modeling environments. Custom channels can be defined by subclassing the Wireless

Channel base class and by attaching functionality defined in Ptolemy II models. It is intended to enable the research community to share models of disjoint aspects of the sensor nets problem and to build models that include sophisticated elements from several aspects.VisualSense is built on top of Ptolemy II, a framework supporting the construction of such domain-specific tools.

A.3 Viptos

Viptos (Visual Sense and TinyOS) is an integrated graphical development and simulation environment for TinyOS-based wireless sensor networks. With this tool, developers can construct TinyOS programs consisting of block and arrow diagrams using a library of existing nesC components. Viptos automatically transforms the diagram into a nesC program that can be compiled and downloaded onto any TinyOS-supported target hardware from within the graphical environment. Viptos allows the developer to graphically simulate networks of wireless sensor nodes running usercreated TinyOS programs.

Viptos is based on TOSSIM and Ptolemy II. TOSSIM is an interrupt- level simulator for TinyOS programs. It runs actual TinyOS code but provides software replacements for the simulated hardware and models network interaction at the bit or packet level. Ptolemy II is a graphical software system for modeling, simulation, and design of concurrent, real-time, embedded systems. Ptolemy II focuses on assembly of concurrent components with well-defined models of computation that govern the interaction between components. VisualSense is a Ptolemy II environment for modeling and simulation of wireless sensor networks at the network level.

Viptos provides a bridge between VisualSense and TOSSIM by providing interrupt-level simulation of actual TinyOS programs, with packet-level simulation of the network, while allowing the developer to use other models of computation available in Ptolemy II for modeling various parts of the system. While TOSSIM only allows simulation of homogeneous networks where each node runs the same program, Viptos

supports simulation of heterogeneous networks where each node may run a different program. Viptos simulations may also include non- TinyOS-based wireless nodes. Viptos allows the user to use different channel models and to change other parts of the simulated environment, such as creating models to generate simulated traffic on the wireless network.

Viptos inherits the actor-oriented modeling environment of Ptolemy II, which allows the developer to use different models of computation at each level of simulation. At the lowest level, Viptos uses the discrete-event scheduler of TOSSIM to model the interaction between the CPU and TinyOS code that runs on it. At the next highest level, Viptos uses the discrete-event scheduler of Ptolemy II to model interaction with mote hardware, such as the radio and sensors. This level is then embedded within VisualSense to allow modeling of the wireless channels to simulate packet loss, corruption, delay, etc. The user can also model and simulate other aspects of the physical environment including those detected by the sensors (e.g., light, temperature, etc.), terrain, etc.

A.3.1 Introduction to Viptos

Wireless sensor networks provide a way to create flexible, automated data collection and monitoring systems. Building sensor networks today requires piecing together a variety of hardware and software components, each with different design methodologies and tools, making it a challenging and error-prone process. Typical networked embedded system software development may require the design and implementation of device drivers, network stack protocols, scheduler services, application-level tasks, and partitioning of tasks across multiple nodes. Little or no integration exists among the tools necessary to create these software components, mostly because the interactions between the programming models are poorly understood. In addition, these tools typically have little infrastructure for building models and interactions that are not part of their original scope or software design paradigms. The goal of this

work is to create integrated tools for networked embedded application developers to model and simulate their algorithms and quickly transition to testing their software on real hardware in the field, while allowing them to use the programming model most appropriate for each part of the system.

We choose to focus on TinyOS, an open-source rutime environment designed for sensor network nodes known as motes, as our underlying programming platform. TinyOS has a large user base over 500 research groups and companies use TinyOS on the Berkeley/Crossbow motes. It has been ported to over a dozen platforms and numerous sensor boards, and new releases see over 10,000 downloads. TinyOS differs from traditional operating system models in that events drive the behavior of the system. Using this type of execution, battery-operated nodes can preserve energy by entering sleep mode when no interesting events are happening.

A TinyOS program consists of a graph of components that are written in an object-oriented style using nesC , an extension to the C programming language. TOSSIM , a TinyOS simulator for the PC, can execute nesC programs designed for a mote. TOSSIM contains a discrete event simulation engine which allows modeling of various hardware and other interrupt events. Although a large community uses TinyOS in simulation to develop and test various algorithms and protocols, they face some key limitations when using the nesC/TinyOS/TOSSIM programming toolsuite. Users may choose from a few built-in radio connectivity models in TOSSIM, but it is difficult to use other models. TOSSIM can efficiently model large homogeneous networks where the same nesC code is run on every simulated node, but does not allow simulation of networks that contain different programs. Additionally, a TinyOS program consists of a graph of mostly preexisting nesC components; users must write their programs in a multi-file, text-based format, even though a graphical block diagram programming environment would be much more intuitive.

To address these problems, we consider VisualSense , a Ptolemy II-based graphical modeling and simulation framework for wireless sensor networks that supports actor-oriented definition of sensor nodes, wireless communication channels, physical media

such as acoustic channels, and wired subsystems. Ptolemy II is a modeling, simulation, and design environment for hierarchical, concurrent, real-time, and embedded systems. VisualSense mainly provides an abstract, mathematically-based modeling environment, and node models must be created from scratch. VisualSense does not provide a mechanism for transitioning from a sensor network application developed within the framework to an implementation for real hardware without rewriting the code from scratch for the target platform.

Integrating TinyOS and VisualSense combines the best of both worlds. TinyOS provides a platform that works on real hardware with a library of components that implement low level routines. VisualSense provides a graphical modeling environment that supports hierarchical, heterogeneous systems. In this report, we present Viptos (Visual Ptolemy and TinyOS), a tool for joint modeling and design of wireless networks and actual sensor node software.

Appendix B

PSOC

B.1 PSOC-Programmable System on Chip

PSoC (Programmable System on Chip) represents a whole new concept in microcontroller development. In addition to all the standard elements of 8-bit microcontrollers, PSoC chips feature digital and analog programmable blocks, which themselves allow implementation of large number of peripherals.

Digital blocks consist of smaller programmable blocks that can be configured to allow different development options. Analog blocks are used for development of analog elements, such as analog filters, comparators, intrumentational (non)inverting amplifiers, as well as AD and DA convertors.

Theres a number of different PSoC families you can base your project upon, depending on the project requirements. Basic difference between PSoC families is the number of available programmable blocks and the number of input/output pins.

Number of components that can be devised is primarily a function of the available programmable blocks. Depending on the microcontroller family, PSoC chips have 416 digital blocks, and 312 analog programmable blocks.

B.2 Block Diagram of PSOC

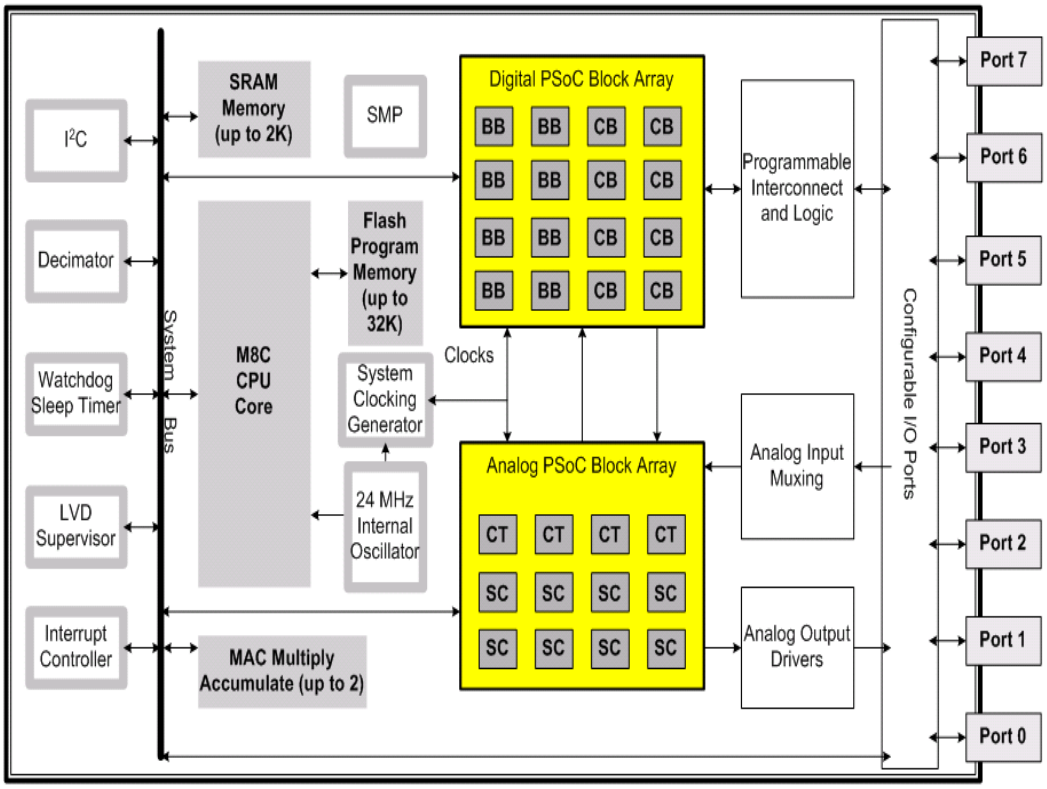

Figure B.1: Basic Blocks of PSOC

B.3 Characteristics of PSOC microcontrollers

Some of the most prominent features of PSOC microcontrollers are:

- MAC unit, hardware 8x8 multiplication, with result stored in 32-bit accumulator,
- Changeable working voltage, 3.3V or 5V,
- Possibility of small voltage supply, to 1V,
- Programmable frequency choice.

Programmable blocks allow to devise:

- 16K bytes of programmable memory.
- 256 bytes of RAM.
- AD convertors with maximum resolution of 14 bits.
- DA convertors with maximum resolution of 9 bits.
- Programmable voltage amplifier.
- Programmable filters and comparators.
- Timers and counters of 8, 16, and 32 bits.
- Pseudorandom sequences and CRC code generators.
- Two Full-Duplex UARTs.
- Multiple SPI devices.
- Option for connection on all output pins.
- Option for block combining.
- Option for programming only the specified memory regions and write protection.
- For every pin there is an option of Pull up, Pull down, High Z, Strong, or Open pin state.
- Possibility of interrupt generation during change of state on any input/output pin.
- IC Slave or Master and Multi-Master up to speed of 400KHz.
- Integrated Supervisory Circuit.
- Built-in precise voltage reference.

Web References

[1] http://www.eecs.berkelly.org

[2] http://www.freemat.org

[3] http://www.scilab.org

[4] http://www.cypress.com

References

[1] Edward A. Lee,Xiaojun Liu, Stephen Neuendorffer,Yang Zhao,Haiyang Zheng "Heterogeneous Concurrent Modeling and Design in Java" April 1, 2008.

[2] Edward A. Lee "Finite State Machines and Modal Models in Ptolemy II" November 1, 2009.

[3] Xiaojun Liu, Jie Liu, Johan Eker, Edward A. Lee "Heterogeneous Modeling and Design of Control Systems" Oct. 27,2001.

[4] Edward A. Lee "Hybrid System Modeling:Operational Semantics Issues" Feb. 4, 2004.

[5] Elaine Cheong,Edward A. Lee, Yang Zhao "Modeling and Design of Wireless Networks and Sensor Node Software" November 17, 2006

[6] Mohamed A. Salem "Ptolemy-Oriented Structural, Reconfigurable,and Heterogeneous Hardware Design,Verification and Synthesis"The Sixth Biennial Ptolemy Conference,Spring 2005

[7] Philip Baldwin, Sanjeev Kohli,Edward A. Lee,Xiaojun Liu,Yang Zhao "Visualsense : Visual Modeling for Wireless and Sensor Network Systems" July 15, 2005

[8] Bilung Lee, Edward A. Lee "Interaction of Finite State Machines and Concurrency Models",November,1998.

[9] Edward A. Lee "Ptolemy Project Status and Overview".

[10] Christopher Brooks, Edward A. Lee, Stavros Tripakis "Exploring Models of Computation with Ptolemy II"

[11] Edward A. Lee, Alan Kamas, Brian L. Evans, "Design and Simulation of Heterogeneous Systems using Ptolemy "

[12] Alain Girault, Bilung Lee, and Edward A. Lee "Hierarchical Finite State Machines with Multiple Concurrency Models",June, 1999.

[13] Philip Baldwin, Sanjeev Kohli, Edward A. Lee "Modeling of Sensor Nets in Ptolemy II" April 26-27, 2004.

[14] Elaine Cheong , Edward A. Lee, Yang Zhao "VIPTOS :Joint Modeling and Design of Wireless Networks and Sensor Node Software",February 15, 2006.

[15] "Psoc basics", Cypress Microsystems. (PPT)

[16] Gang Zhou, Man-Kit Leung, and Edward A. Lee,"A Code Generation Framework for Actor-Oriented Models with Partial Evaluation"

[17] "CY3209-ExpressEVK Quick Start Guide",Cypress Microsystems.

[18] Matt Welsh,"nesC: A component-oriented language for networked embedded systems", Intel Research Berkeley and Harvard University.

[19] David Gay, Philip Levis, David Culler,"nesC 1.1 Language Reference Manual ", May 2003

[20] "PSoCTM Designer: C Language Compiler",User Guide,Cypress MicroSystems, Inc.

[21] "Algorithm - Embedded State Machine Design for PSoC using 'C' Programming (Part I , II , III)",Cypress Microsystems.