**EE 552**
**CDMA Based Communication System**
**Final Report**

**Group Members:**
**Xiaofei Vivien Dong:** dongx@ee.ualberta.ca
**Ai Hua :** hai@ualberta.ca
**Gautam Karnik:** gkarnik@ualberta.ca

Dr. Elliott
Lec B1
April 3, 2001

## Part 1. Declaration of Original Content

The design elements of this project and report are entirely the original work of the authors and have not been submitted for credit in any other course except as follows:
LCD driver “musr_LCD_control.vhd” was modified from [14]
Keypad driver "musr_keypad.vhd" was modified from [13]

Signed by: Xiaofei Dong Ai Hua Gautam Karnik

## Part 2. Abstract

The communication system designed by our CDMA group is capable of multiple user communication with some CDMA characteristics. Multiple users can communicate with a base station utilizing a keyboard and a LCD. A communication base station is built to acknowledge channel access request, maintain the communication process, and switch user frames to the correct destination. Spreading and de-spreading mathematics are adopted and modified from 3GPP Technical Specification of Spreading and Modulation (3GPP TS213) to build and maintain transmission channels between users and base station. The whole system operates correctly in simulation as well as on FPGAs. This final reports gives an overview of the system, details the design strategy and discusses results of experiments we carried out. The design verification and documentation is given in the appendix.

## Part 3. Achievements and System Overview

We have implemented a complete communication system consisting of two users and one base station, with wired transmission media into two Altera UP1 boards. We were able to include some CDMA characteristics in the data transmission and reception procedures.

We have been able to implement all the functions we proposed in the Project Proposal. The system is capable of:

- accepting one user input data from key pad;
- encoding user keypad data;
- generating transmit frame according to the simplified CDMA air interface protocol
- spread-spectrum communication at spreading factor 8 (every one bit information is scrambled onto 8-bit PN code)
- frame synchronization to synchronize transmission and reception
- de-spreading and recovering information from the PN sequences
- simple processing on received frames
- LCD and DIGIT display of received information
- maintaining the communication channel between mobile user and base station
- accommodating two users

We use one UP1 board as the mobile users, the other one as the base station. Mobile user board is connected to keypad and LCD, and the base station board is connected to the LEDs and digit display. Both chips function correctly in simulation. Both boards function correctly when connected with peripherals and the hardware is fully tested. The system operates at 25MHz, the total logic cells used for mobile user board is 648 (56%), for base station is 296 (25%).

**Mobile User board**

The mobile user board is capable of:

- accommodating two users
- accepting input data from the keypad
- encoding the keypad input mobile user data
- creating a data frame with user data and channel code
- spreading the data over a fixed 8-bit fixed PN sequence code
- attaching a synchronization code to the frame and sending the information to the base station
- receiving spread data from the Base Station Board
- synchronizing to the data
- de-spreading the data frame
- analyzing the data frame
- displaying the user data on the user's LCD

**Base Station Board**

The base station is capable of:

- receiving spread data from the Mobile User Board
- synchronizing to the data
- de-spreading the data frame
- analyzing the data frame with user data and channel code
- displaying the user data on the FLEX DIGIT display
- building a response frame
- sending the response frame to the Mobile User Board

**Possible future work and add-on functions**
The design of the system can be potentially upgraded to allow multiple users to transmit simultaneously because the central processing units are flexible. To upgrade to multiple users transmitting simultaneously:

- more input devices are needed
- each input device has its own encoder like the keypad encoder

Multiple users can share the rest of the components.

In the previous documents involved with this project we discussed the possibility of using infrared transceivers as the physical medium for transferring data between mobile users and the base station. We were not able to implement this feature due to time constraints.

We have been able to simulate and test the back-end and interface components in this communication system and we are confident that we will be able to have the physical system implemented.

## Part 4. Description of Operation

For the third generation mobile radio communication system (known as 3G), code division multiple access (CDMA) has become the mainstream air interface. Our design is a simplified CDMA communication system, focusing on the most prominent feature of a CDMA mobile system-spread spectrum communication. The goal of our design is to implement the system with wired connection between the mobile users and the base station. We adopt CDMA air interface design strategies in our protocol, frame design as well as data processing design, so we can

easily proceed to the more complicated design stage of a real multiple user wireless communication system.

This section describes the layout, data flow, frame format for data transmission and the mathematical algorithms used in this system. It also includes information on the basics of CDMA systems and what concepts of CDMA system we have adopted.

**Physical Layout of System**

This system utilizes 2 FLEX 10K UP1 Boards. One UP1 board connected with a keypad and 2 LCDs serves to simulate 2 simultaneous mobile users. The keypad is divided into 2 horizontal sections corresponding to each user. Another UP1 board serves as a base station to acknowledge the channel access request from users, maintain the communication channel, echo the mobile user data and display user input remotely.

The mobile user board and base station board will be connected to each other using the FLEX_EXPAN ports on each board. The base station will output each user's data to the digital display on the board. Below is a diagram of how our system will look physically:

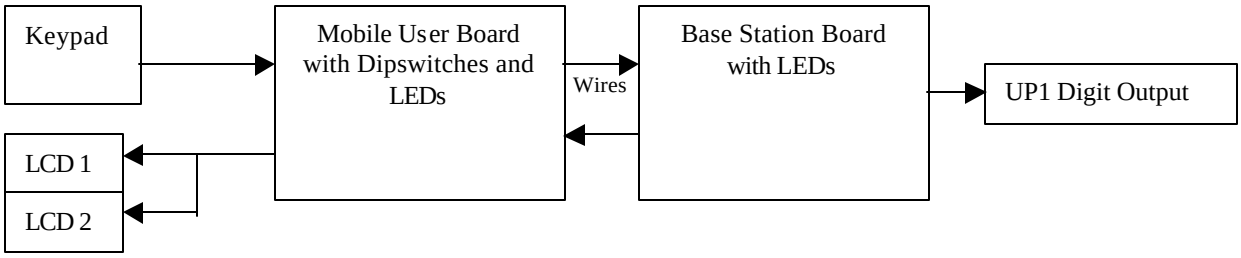



Fig. 0

**Data Frame Format**

To make our design more flexible for upgrading to a real CDMA system, we design our interface functions into protocol layers similar to the air interface protocol in IS-95. The layered structure of CDMA air interface is as follows:

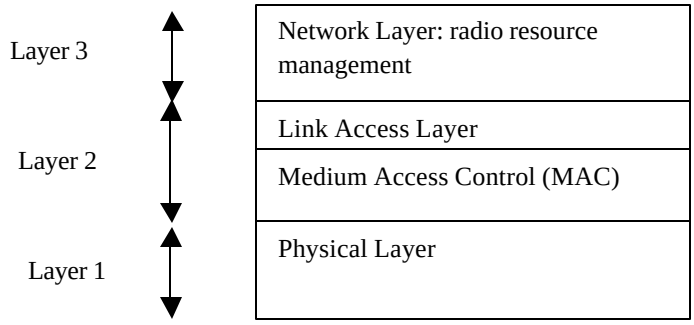



Fig. 1

We will use the following frame structure for data transmission:

| Synchronization | Channel header(function code) | Data |
|---|---|---|

Fig. 2

The protocol layers to generate this frame is illustrated as:

| Link Access layer: generate sync code, generate frame |
|---|
| MAC: Assemble user data, Channel header |
| Physical layer: spreading |

Fig. 3

The sync code will be 8 bits long and will be defined as 0x72 (see next sub-section for more details). The channel header will be 4 bits long and will have the following types of values:

| | |
|---|---|
| Random Access Request | 0011 |
| Acknowledge | 1100 |
| Communication Send Mobile User | 1010 |
| Communication Send Base Station | 0101 |
| Close Frame | 0111 |

Fig. 4

The data section of this frame will be used in order to exchange data between users and base station. The structure of user data package is defined as below:

| Data User 1 | | Data User 2 | |
|---|---|---|---|
| On/off | 3bits user data | On/off | 3bits user data |

Fig. 5

The Data User 1 and Data User 2 section will each be 4 bits long and will contain 1 function bit and 3 data bits sent from the different users. The function bit is used to indicate if the user is communicating or terminating connection to base station. The content of the 3-bit data portion is related to the 8 keys on the keypad designated to the specific user. In total the entire communication frame will be 20 bits long. In the Physical Layer, spreading will occur after the synchronization code to the end of the frame.

**Synchronization Code Allocation**

We use 1 don't care bit plus a 7-bit Bark Code as our synchronization code. The 7-bit Bark Code used for our design is {1, 1, 1, 0, 0, 1, 0}. The 1 don't care bit plus the Bark code translates into 0x72 for our synchronization code. This Bark code has a sharp auto-correlation function. If we use a Bark Code correlator as our synchronization code detector, the probability of false synchronization or missing one frame is small. This Bark code is not sensitive to the phase shift, so it is suitable for use in synchronization.

The Bark Code Detector has the following structure:

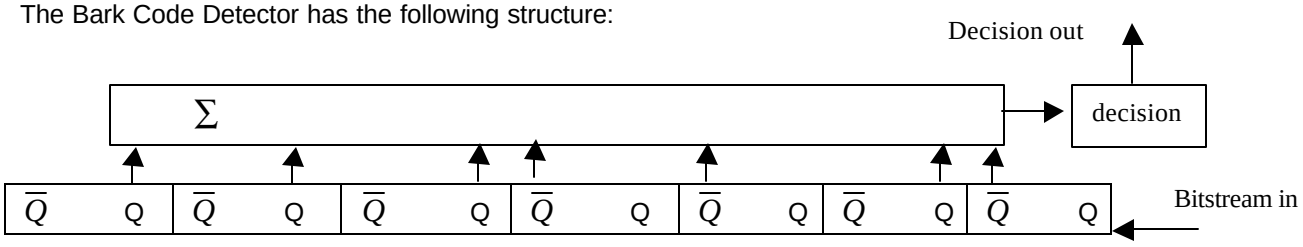



Fig. 6 Bark Code Detector

**Spreading and De-spreading Algorithms**

CDMA Concepts

In CDMA each user is assigned a unique code sequence (spreading code) it uses to encode its information -bearing signal [1]. The receiver, knowing the code sequences of the user, decodes a received signal after reception and recovers the original data. The spectral spreading of the transmitted signal gives to CDMA its multiple access capability. The benefits of the spreading the signals are:

- Multiple access capability: the receiver can recover a user's data if each user has a unique code from a code family that has a low cross-correlation.
- Protection against multi-path interference
- Privacy: the transmitted signal can only be de-spread if the code is known to the receiver
- Interference rejection: background noise power is reduced after de-spreading

Spread-Spectrum Algorithm

There are different multiple access schemes for spreading the user data. In our design we use the direct sequence spreading (DS-CDMA). In DS-CDMA [1], [2], the data signal is directly multiplied by a digital code signal.

Spreading Code Selection Criteria

The basic properties of spreading codes are that they have low cross-correlation between the desired and interfering users, and good auto correlation properties to separate the multi-path components. The auto correlation and cross-correlation are connected in a way that it is not possible to achieve good auto correlation and cross-correlation simultaneously.

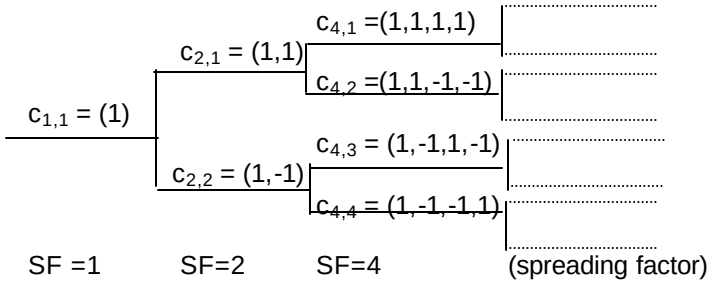
A spreading code can be short or long. Path delay, the number of users, cell size, etc. are all factors that influence the selection of short or long code. In our system, we have small number of users, short delay path, so short spreading code is used.

When the number of users is small, and the codes are very short, we focus more on the maximum correlation property when choosing the spreading code to minimize the maximum peak even cross-correlation. Orthogonal codes like Walsh codes are used in our design.

Walsh codes can be constructed according to [2, eqn. 5.1],

$$H_0 = [0] \qquad H_n = \begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & \overline{H}_{n-1} \end{bmatrix}$$

and the resulting tree-structured orthogonal codes are:




In our design, SF= 8 is used. The spreading code for current design with one channel is $W_1$ = {1,1,1,1,0,0,0,0}.

Spreading Algorithm

Before spreading, both user input data and spreading code {1, 0} is mapped to {1, -1}. The zeros do not mean the absence of signals. Instead, it means the shift of signal phase of 180 degree.
Every input bit {0| or {1} is multiplied by $W_1$. In our simplified spreading algorithm, bit '0' becomes "11110000", while data '1' becomes "00001111". Suppose the bit rate of the user data is R bps, after spreading, the bit rate is 8*R bps.

De-spreading algorithm

After the synchronization code is detected, 8 bits of the input data is buffered for de-spreading. The 8-bit buffer stream is fed into an 8-bit correlation detector containing a correlator bank, which has a similar structure as the Bark Code Detector in Fig. 6. After the decision device, bit pattern {11110000} is interpreted as '0', and {00001111} is interpreted as '1'. The output bit rate after the de-spreading will be R.

Walsh Correlation Decoding

Let two sequences be given by
$A = (a_1, a_2, \ldots a_N)$, and
$B = (b_1, b_2, \ldots b_N)$

The correlation measure is calculated by adding up all the term-by-term products, [2, eqn. 5.35a]

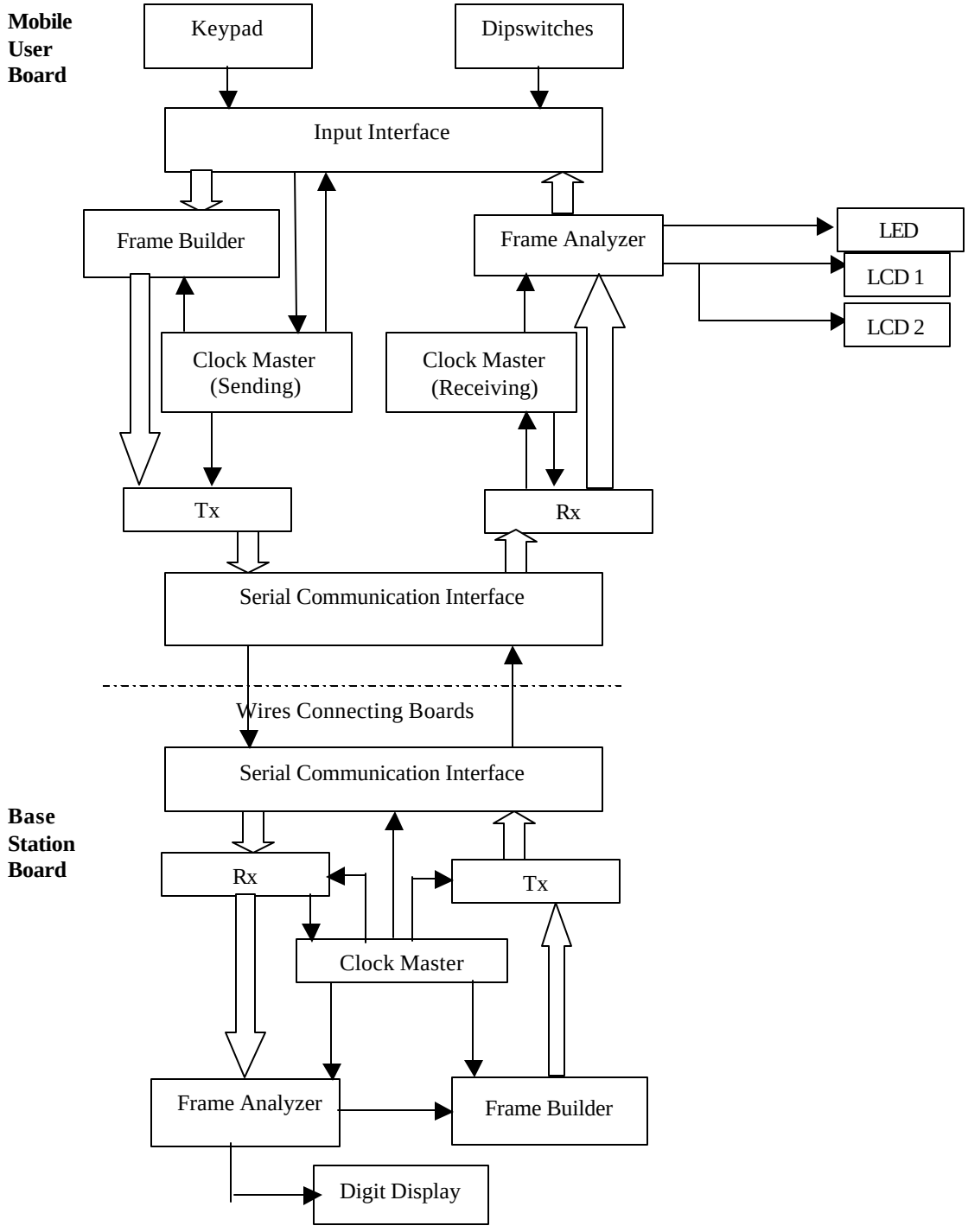
If we define the "weight" of a sequence as the number of 1s in the sequence, then the above concept can be expressed as

$\langle A,B \rangle$ = (number of term-by-term agreements in A and B) –
( number of term-by-term disagreements between A and B)
= N - 2 * weight( A XOR B)

**System Layout**

The Wireless Communication Base Station will have the following system layout:

Fig. 7 System Structure



The input interfaces will be responsible for communicating with peripheral components that the user can interact with. The serial communication interface will be responsible for transport of data between the two UP1 boards. The transmitter will be responsible of spreading the signal before the data is sent. The receiver will be responsible for frame synchronization and de-spreading the signal before data is received. The timing controller—Clock Master as in our design, will co-ordinate the timing relationship with other parts of the system to initialize, to maintain, and to terminate the communication between users and base station. The frame builder and frame analyzer are responsible for assembling the data frame and disassemble the frame.

**System Data Flow**

The basic data flow of this system can be divided into three main components:

1. Initialization
2. Communication
3. Closing Communication

1. Initialization

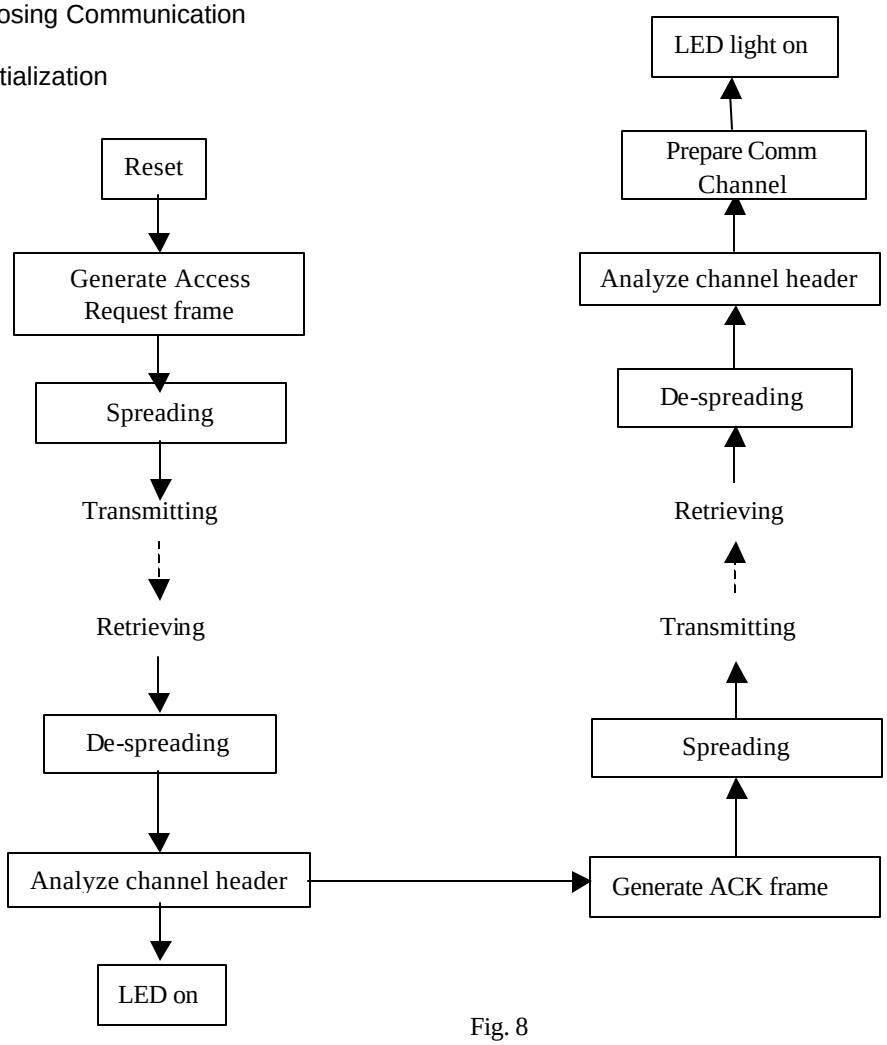



Fig. 8

When one user dipswitch is turned on, that user is sending a channel access request. A frame is constructed with the Random Access Request code in the Channel Header. Each bit after the sync code is spread and the frame is transmitted from the mobile user board to the base station board. When the base station board receives the frame, it must first find the sync code. Once this is accomplished, the rest of the data frame is de-spreaded. The channel header is analyzed and if it recognizes the Random Access Request, an LED on the base station lights and another frame is constructed with the Acknowledge code in the Channel Header. Each bit after the sync code is spreaded and the frame is transmitted from the base station board to the mobile user board. When the mobile user board receives the frame, it finds the sync code, and de-spreads the rest of the frame. If the Acknowledge signal is recognized in the frame, an LED is lit on the mobile user board, the channel enters communication mode and is ready to receive keypad input and transmission.

2. Communication

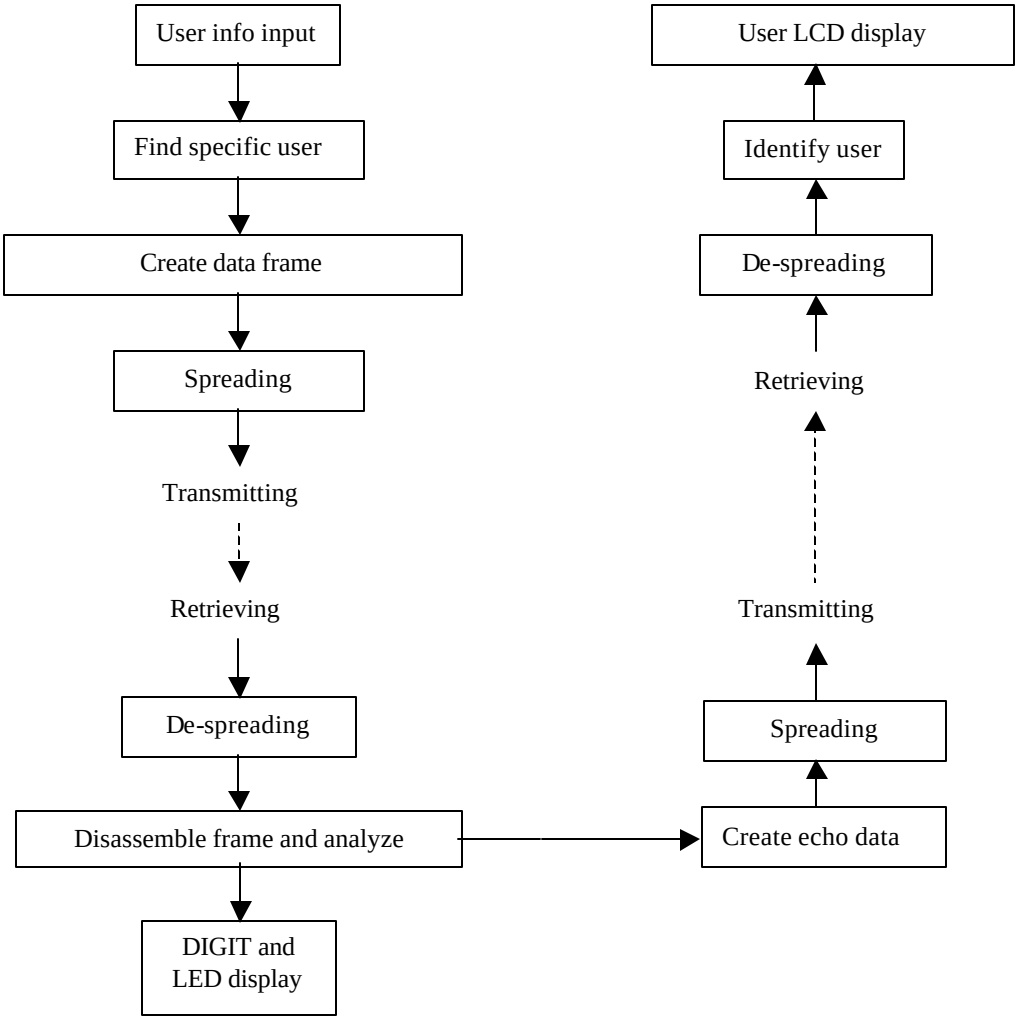
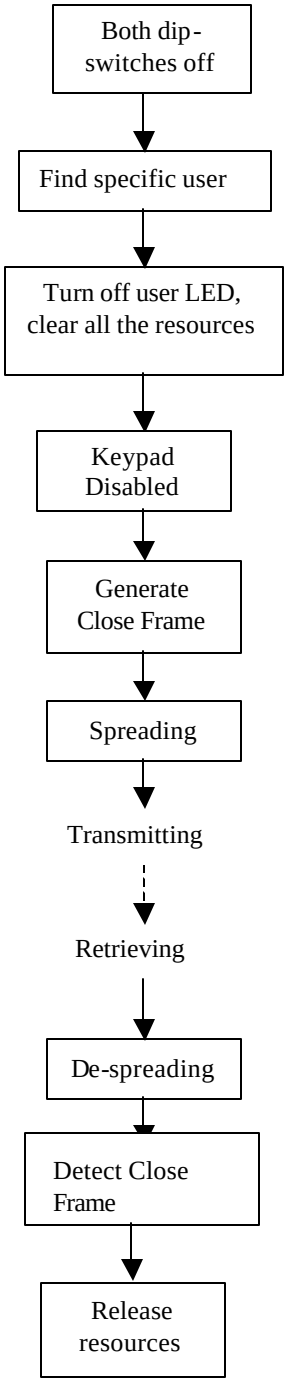



Fig. 9

When the user presses a key on his keypad, a frame is created containing the Communication Send code in the Channel Header, and the user data information. Each bit after the sync code spreads over the scrambling code and the data frame is transmitted from the mobile station board to the base station board. When the base station board receives the frame, it finds the sync code, and de-spreads the rest of the frame. The frame is disassembled and analyzed. If the Communication Send code is detected, the data portion of the frame is displayed on the FLEX_DIGIT display. Another frame is constructed with the Communication Send code in the Channel Header and the user data is echoed in the data portion of the frame. Each bit after the sync code is spread and the data frame is transmitted from the base station board to the mobile user board. When the mobile user board receives the frame, it finds the sync code, and de-spreads the rest of the frame. If the Communication Send signal is recognized in the frame, the LCD for the particular user displays the data.

3. Closing Communication




The user turns off their switch to terminate their connection. The user's LED is extinguished on the mobile user board, the LCD is cleared and the keypad is no longer functional for the specified user. A close frame is generated in the mobile user board when both users are offline, spreads and then transmitted to the base station. The base station de-spreads the frame, analyzes the frame and releases any system resources and clears any display output.

## Part 5. Datasheet for FPGA

### System Hardware

The communication system designed involves two Altera UP1 boards to serve as mobile users and base station respectively. The FPGA chips used on the two boards are FLEX10K20RC240-4. The whole system hardware includes one keypad, two LCDs, two UP-1 boards, ± 5V power supply, 4 10kΩ resistors, 2 10kΩ variable resistors, and one breadboard.

### Features

- Working frequency of the system is 25MHz
- Operating on 5V DC power supply
- Maximum clock frequency of system is 29.33MHz
- Two users share the transmission media
- Wired connection between mobile users and base station
- Transmission/reception bit rate 6.25M bps
- Information bit rate 781,25Kbps
- Spread-spectrum communication at spreading factor 8
- Changeable synchronization code—7-bit Bark code {1110010} used in current design
- Changeable short spreading code—8-bit Walsh code {11110000} for both users in current design
- Original frame structure according to CDMA air interface protocol

### General Description

The mobile user board collects user input data from the keypad, which is divided horizontally into two parts for two users. The mobile user board assembles frame according to the frame structure defined in Part 4 Description of Operation. It further spreads data onto an 8-bit scrambling code and transmits the bit stream at bit rate 6.25Mbps. The base station samples the received bit stream at 4 times the bit rate at 25MHz. The base station detects the frame synchronization, de-spreads data and analyzes the frame. Useful data are sent to LCD display. Channel control information is collected, assembled as a frame, spreads onto the scrambling code and transmitted from the base station back to the mobile user. The mobile user board is capable of frame synchronization with the base station and processing of the coded bit streams. Both the mobile user and the base station chip maintain the communication channel for correct operation.

### FPGA IO PINS

Mobile User Board

| Name | Direction | Pin Number | Description |
|---|---|---|---|
| Clock | Input | 91 | System Clock (25MHz) |
| Reset | Input | 28 | Active low |
| KP_ROW (0...3) | Output | FLEX_EXPAN-A 50, 51, 53, 54 | Keypad response |
| KP_COL (0...3) | Input | FLEX_EXPAN-A 61-64 | Keypad response |
| LCD1_Enable | Output | FLEX_EXPAN-A 70 | LCD User 1 Enable |
| LCD1_RW | Output | FLEX_EXPAN-A 72 | LCD User 1 R (1) / W (0) |
| LCD1_Select | Output | FLEX_EXPAN-A 74 | LCD User 1 Control Instructions |

| | | | |
|---|---|---|---|
| LCD1_Data (0...7) | Output | FLEX_EXPAN-A 76, 79, 81,82, 80, 78, 75, 73 | LCD User 1 Data |
| LCD2_Enable | Output | FLEX_EXPAN-A 83 | LCD User 2 Enable |
| LCD2_RW | Output | FLEX_EXPAN-A 86 | LCD User 2 R (1) / W (0) |
| LCD2_Select | Output | FLEX_EXPAN-A 88 | LCD User 2 Control Instructions |
| LCD2_Data (0...7) | Output | FLEX_EXPAN-A 95, 98, 100, 101, 99, 97, 94, 87 | LCD User 2 Data |
| FLEX_SWITCH-1 | Input | 33 | User 1 Request / Termination |
| FLEX_SWITCH-2 | Input | 41 | User 2 Request / Termination |
| User 1 Connection | Input/Output | FLEX_EXPAN-A 45 wired to LED Female Header Positions 1 | User 1 Connection Status Indicator |
| User 2 Connection | Input/Output | FLEX_EXPAN-A 46 wired to LED Female Header Positions 2 | User 2 Connection Status Indicator |
| Channel Ready | Output | FLEX_EXPAN-A 48 wired to LED Female Header Positions 8 | Channel Status Indicator |
| Serial Transmitter Port Data | Output | FLEX_EXPAN-B 162 | Data Transfer |
| Serial Receiver Port Data | Input | FLEX_EXPAN-B 157 | Data Transfer |

Table 1. FPGA Pin Estimation for Mobile User

Total pin number is 39.

Base Station Board

| Name | Direction | Pin Number | Description |
|---|---|---|---|
| Clock | Input | 91 | System Clock |
| Reset | Input | 28 | Active low |
| FLEX_DIGIT | Output | 6-9, 11-13, 17-24 | User data display |
| Serial Transmitter Port Data | Output | FLEX_EXPAN-C 226 | Data Transfer |
| Serial Receiver Port Data | Input | FLEX_EXPAN-C 230 | Data Transfer |

Table 2. FPGA Pin Estimation for Base Station

Total pin number is about 18.

**Parameters**

| Name | Typical Value | Customization |
|---|---|---|
| Number of users | 2 | More provided more keypads and peripheral processor |
| User data length | 3 | Changeable with small modification of the timing controller and core processor |
| Frame length | 12 | Changeable with small modification of the core processors |
| Synchronization code | 8 | Changeable with small modification of synchronizer and timing controller |
| Scrambling code | 8 | Changeable with small modification of the timing controller and spreader/de-spreader |

**Performance**

| Implementation | Speed (Clock) | Maximum Achievable Speed (clock) | Logic Cells | EABs |
|---|---|---|---|---|
| Mobile User Chip | 25MHz | 27.39MHz | 648/1152 56% | 0 |
| Base Station Chip | 25MHz | 48.78MHz | 296/1152 25% | 0 |

## Part 6. FPGA Logic Blocks Estimation/Measurement

The system contains altogether 18 major logic modules. On the mobile user side, 10 logic modules are declared and on the base station side 8 are needed. Logic modules vary in size and speed, and may contain many components and smaller logic modules. The following table shows how many logic cells are needed for all major logic modules and how the data is obtained (measurement of estimation). All of the modules have been compiled, only those that have been simulated and can function as required are considered complete and the data are given as measured from the MaxPlus2 .rpt file.

***Table 3: Logic Block Requirement for Mobile User***

| Modules | Sub-components contained | Logic Cells Needed | Maximum clock frequency | Source of Data |
|---|---|---|---|---|
| Musr_chip (top level) | All the mobile user sub-components | 648 (56%) | 27.39MHz | Measured |
| Musr_switch | None | 4 (0%) | Max delay = 12.8ns (Comb. Logic) | Measured |
| Musr_clk_master | None | 44 (3%) | 36.36MHz | Measured |
| Musr_receiver_ clk_master | None | 36 (3%) | 47.61MHz | Measured |
| Musr_keypad | KeyDecode KeyEncode | 134 (11%) | 26.31MHz | Measured |

| | | | | |
|---|---|---|---|---|
| | KeyDriver<br>KeySense<br>KeyDebounce<br>KeyValidator<br>Registers | | | |
| Musr_frame_builder | None | 25 (2%) | 95.23MHz | Measured |
| Musr_transmitter | Parallel_to_serial converter, modular 2 operator, spreading code register, synchronization code register, output data latch | 56 (4%) | 90.90MHz | Measured |
| Musr_synchronizer | Serial_to_parallel converter, correlation operator, summing network, decision unit, pipeline register | 64 (5%) | 59.92MHz | Measured |
| Musr_despreader | Spreading code register, correlation network, data latch | 15 (1%) | 78.12 MHz | Measured |
| Musr_frame_analyser | None | 37 (3%) | 100MHz | Measured |
| Musr_LCD_control | LCD Driver * 2<br>Counter | 238 (20%) | 24.93 MHz | Measured |

***Table 4. FPGA Logic Blocks Requirement for Base Station***

| **Modules** | **Sub-components contained** | **Logic Cells Needed** | **Maximum Clock Frequency** | **Source of Data** |
|---|---|---|---|---|
| Base_chip (top level) | All the base station Sub-component | 296 (25%) | 45.24MHz | Measured |
| Base_clk_master | None | 44 (3%) | 36.36 MHz | Measured |
| Base_receiver_ clk_master | None | 36 (3%) | 47.61 MHz | Measured |
| Base_synchronizer | Serial_to_parallel converter, correlationoperator, summing network, decision unit, pipeline register | 64 (5%) | 59.92MHz | Measured |
| Base_despreader | Spreading code register, correlation network, data | 15 (1%) | 78.12 MHz | Measured |

| | | | | |
|---|---|---|---|---|
| | latch | | | |
| Base_frame_analyzer | None | 40 (3%) | 80 MHz | Measured |
| Base_frame_builder | None | 12 (1%) | 125 MHz | Measured |
| Base_digit_control | None | 40 (3%) | 125MHz | Measured |
| Base_transmitter | Parallel_to_serial converter, modular 2 operator, spreading code register, synchronization code register, output data latch | 56 (4%) | 90.90 MHz | Measured |

**Conclusion:**

The system designed can fully function at the maximum clock frequency provided by the hardware. The maximum possible speed for the mobile user board is 27.39MHz, and 48.78MHz for the base station board. The resource requirement for the mobile user board is 648/1152 logic cells (56%), and 296/1152 (25%) for the base station chip. Thus another contribution of the current design is that it has enough resources to upgrade the system and to accommodate multiple users.

## Part 7 Results of Experiments and Characterization

The design scheme is top-down following the hierarchy tree. Speed, resource, stability, and testability are the four main concerns in our design experiments. Centralized control is adopted here rather than distributed control to meet resource requirement. Pipelines are used in some components to meet the speed requirement. Only one globe system clock and one globe reset are used in the design for the consideration of stability. Followed are the experiments and characterization of some typical components declared in Table 3 and Table 4.

**Clock Master**

The first attempt of our design is a distributed system. Components communicate by using handshake signals. The advantage of distributed control includes:

- The timing inside and between components is simple and controlled by a group of handshake signals.
- The circuits can operate continuously.

The disadvantages of distributed control are

- System complexity is high. Every components must have a handshake scheme, thus it has to keep track of current work state. Control units and control signals must be dedicated for handshaking, thus increase the number of logic blocks in design substantially.
- More efforts are needed to keep the speed of the individual components.
- Hard to test individual components.

The main concern of our design is limited resource. Thus in the current design, we use centralized control. In this scheme, most of the processes are triggered by two deployment centers. All the data flows are controlled by the deployment center as well.

The centralized control scheme decreased the system requirements of logic block tremendously. When the control processes are integrated into one or two components, many resources can be

shared. It also simplifies the design of timing for each component. Further more, the system is easy to test. Because almost all the timing relationships inside the chip are integrated into two components, it is easy to simulate, modified and debug the system.

Total logic cells needed by the clock master for sending frame are 40, the maximum clock frequency is 50.5Mhz. Total logic cells needed by the clock master for receiving frame are 33, the maximum clock frequency is 46.29Mhz.

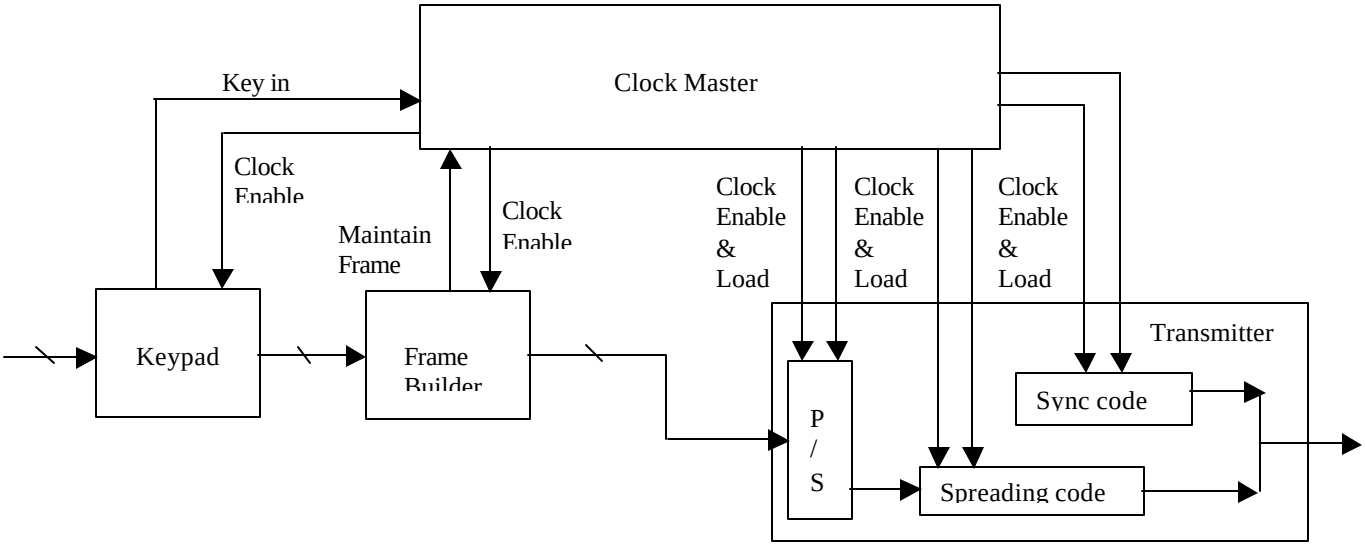
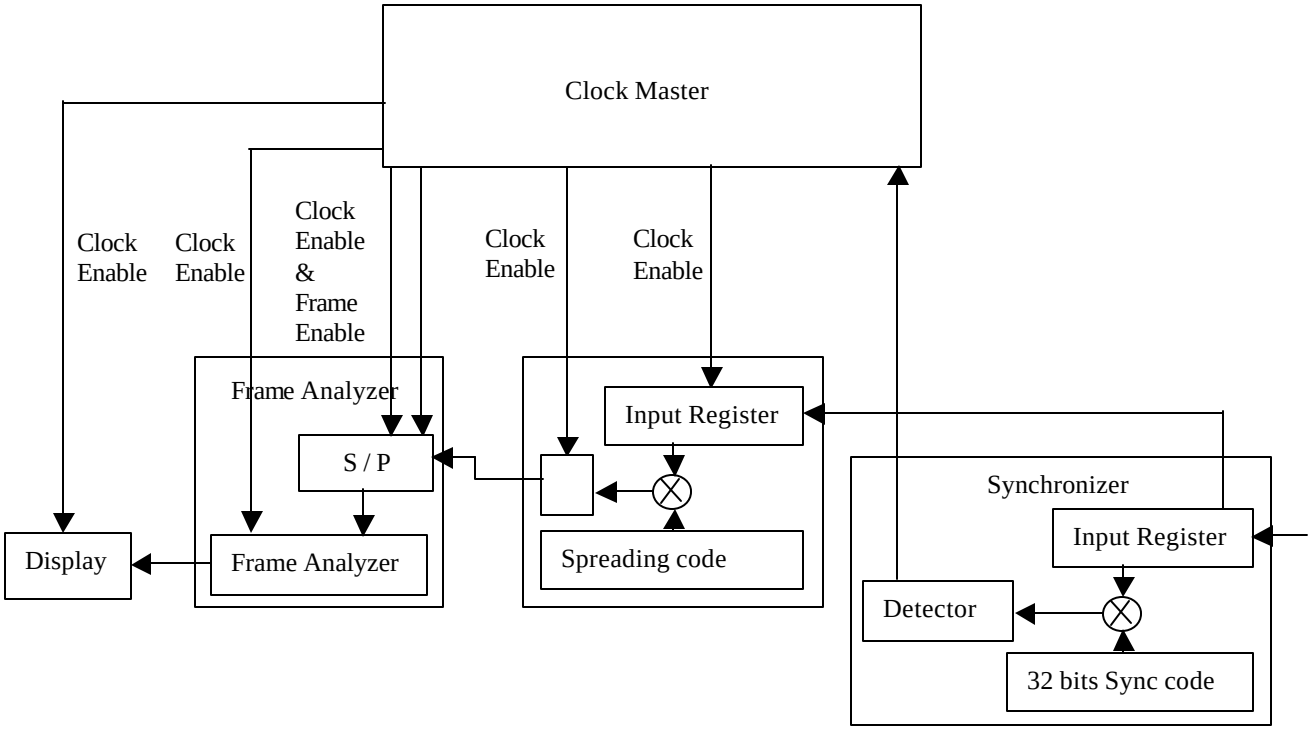



Fig. 1 The Data flow of sending a frame

Below is a figure to demonstrate the process control to receive a frame.



**Synchronizer**

The synchronizer used in our system for frame synchronization is a binary pattern correlator, which compares the input digital data with the preset synchronization code (binary pattern). The correlator contains:

- A shift register that converts the serial input bits into parallel data
- Synchronization code (reference pattern) register that stores the synchronization code
- Correlation unit that correlates the received data with the reference pattern, output a weight vector
- Summing network that finds the distance of the input data and the reference code
- Decision device that makes the decision of which logic data have been received according to the result of the summing network.

The bottleneck of the design for this synchronizer is the summing network, which finds the number of '1's in the weight vector.

The original design is to use the '+' operation in VHDL, i.e. use LPM adders for the summation. In a *for* loop, every bit of the weight vector is summed. The total number of logic cells used for this design scheme is 183 and the minimum clock period is 76.6 ns, which is much longer than the system clock period at 40 ns. To be able to communicate with other modules correctly and keep the timing of the system consistent, the synchronizer must be able to operation at 25 MHz.

We notice that the function can be implemented by several stages of adder banks. The first stage is a half-adder bank of 16 half-adders. Then eight 2-bit adders can be used to sum the adjacent output of the half-adder bank. A 24-bit vector is the output of the 2-bit adder bank, if we stack the output of every adder. Similarly, the third stage adder bank can be built from four 3-bit adders, and the fourth stage is composed of two 4-bit adders, then the last stage is a 5-bit adder. The output of the summing network is a 6-bit vector.

By building the adder banks, we are able to insert pipeline registers after each stage to speed up the summation. The design can be further simplified because only part of the inputs combinations are possible at each adder.

We denote the new adder structure as the incomplete adder. An example for the 3-bit incomplete adder is shown below.

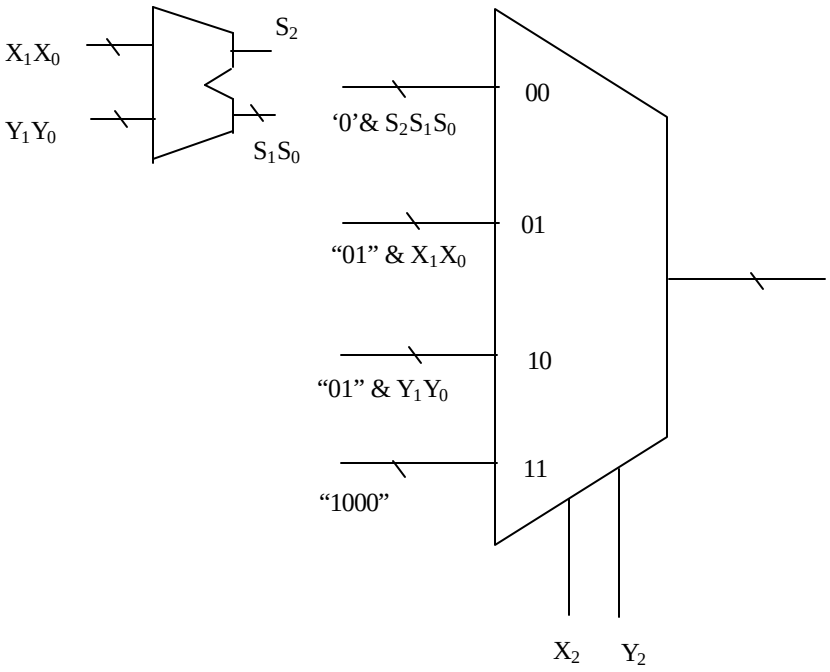
Suppose the input addends to a 3-bit incomplete adders are { $x_2$ $x_1x_0$} and {$y_2y_1y_0$}. { $x_2$ $x_1x_0$} are the outputs from a 2-bit incomplete adder of the previous stage. $x_2$ is the carry and $x_1x_0$ are the sum. They show how many ones are in the 4-bit section of the weight vector. Thus, the possible combination of { $x_2$ $x_1x_0$} are X"0" to X"4". We can use the MSB of vectors x and y as the selector of the multiplexer. When $x_2$ $y_2$ = "00", lower bits of x and y sum like a conventional 2-bit ripple carry adder, and the carry of the 3-bit incomplete adder is zero. When $x_2$ $y_2$ = "01" or "10", value 4 is summed with a vector that is less than 4, so the output carry of the adder is zero, the most significant of the sum part is '1', and the lower bits of the sum part is just the addend that is not four. When $x_2$ $y_2$ = "11", all other bits of x and y must be zeros, so the output is eight.

The speed for this design scheme (without pipeline registers) is 46.6 ns as the minimum clock period. To meet the speed requirement of the system, four pipeline stages are used, each after the adder bank stage. The maximum clock frequency using incomplete adders and pipeline stages is 59.92 MHz. The logic blocks required for the summing network are 155 logic cells.

In the frame synchronizer, using the constant value for the synchronization code, instead of synchronization code register, can further reduce the resource required. The logic blocks required at the top level synchronizer is 64.

Notice that we cannot use one LPM adder with pipelines to implement the summing network. After the half-adder bank, we need to add sixteen 2-bit addends, instead of two 16-bit addends. In this case, the benefit of using pipelined LPM adders comes only from the fact it can speed up the summing at each stage. The trade off is that LPM adder banks consume much more resources. What is more, as discussed earlier in this section, it is a waste of resources to use conventional adders because of the characteristics of the inputs to the adders.

Fig. 2 The structure of a 3-bit incomplete adder




**Part 8 IC test measurements**

This system does not require any major extra circuitry. Only the Mobile User Board has some simple interface circuitry, namely the keypad and LCD contrast control. The keypad requires the use of four 10-kΩ pull-up resistors and the LCD contrast control utilizes a 10-kΩ variable resistor. Please see Appendix B for Data Sheets and Schematics of the keypad and LCD.

## Appendix A References


1. Tero Ojanpera, Ramjee Prasad, "Wideband CDMA for Third Generation Mobile Communications", Artech House 1998

2. Jhong Sam Lee, Leonard E. Miller, "CDMA Systems Engineering Handbook", Artech House, 1998.

3. 3rd Generation Partnership Project: Technical Specification Group Radio Access Network; Spreading and Modulation (FDD), 1999

4. 3rd Generation Partnership Project: Technical Specification Group Radio Access Network; Physical Layer-General Description, 1999

5. C. McCarthy, S. Somogyi, J. Smith, K. Rudko, S. Durand, "Wired CDMA Network", EE552 Course Project, fall 2000.

6. http://www.3gpp.org

7. W-CDMA Solutions; http://www.altera.com /html/mega/solutions/s-alt-wcdma.html

8. CDMA/PCS Library; http://www.elanix.com/ds_cdma.htm

9. Index of /support/example_files/cdma; http://www.elanix.com/support/example_files/cdma/

10. CDMA; http://www.ieee.org/web/developers/webthes/00000967.htm

11. AN 129: Implementing a W-CDMA System with Altera Devices & IP Functions; http://www.altera.com/document/an/an129.pdf

12. Base band Pulse Shaping Filter Abbreviation: Baseband Synopsis; http://www.elanix.com/support/manuals/Cdma.pdf

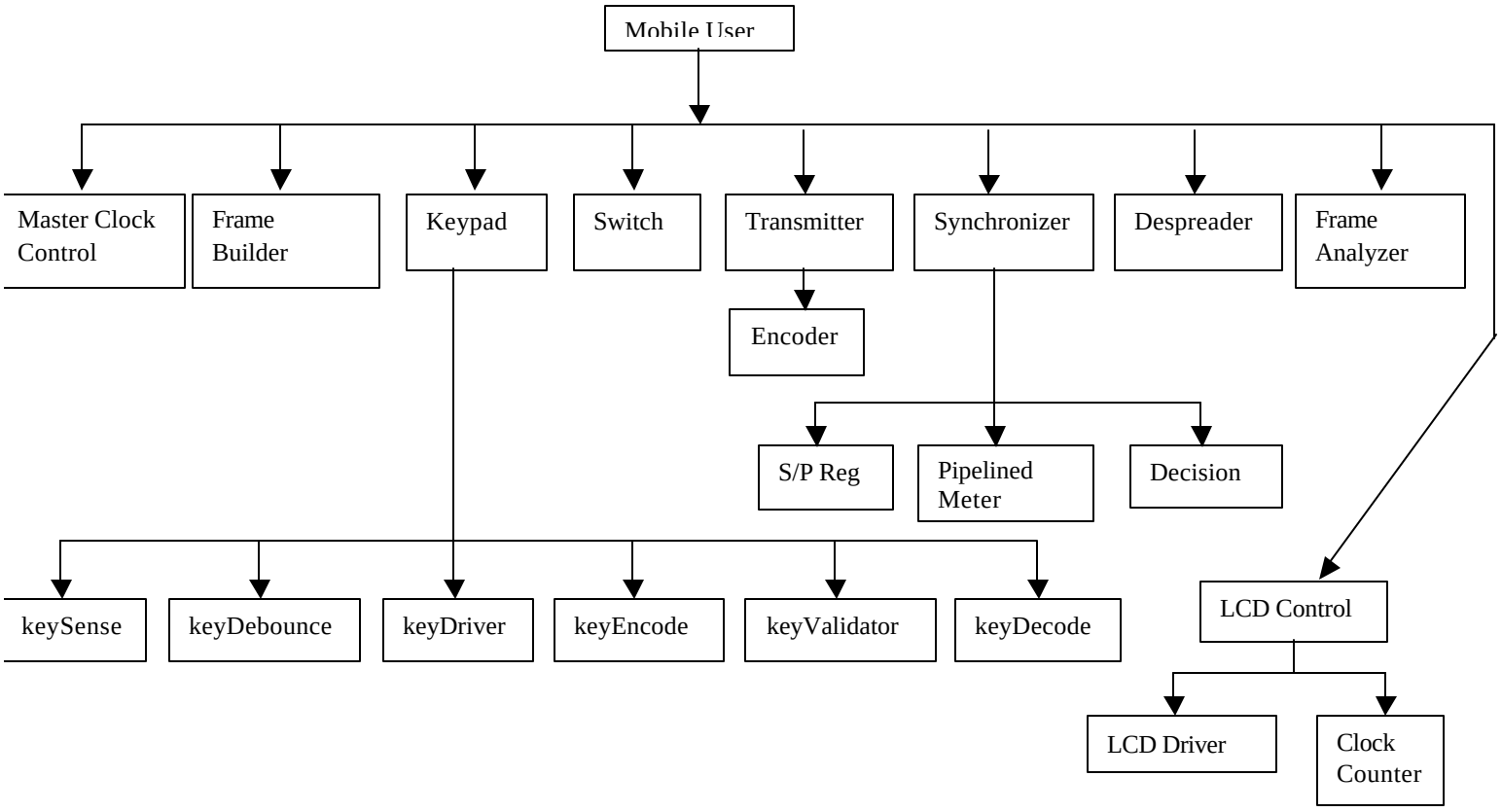
13. C.Wickman "Imaginary Calculator ", EE552 Course Project, winter 1998.

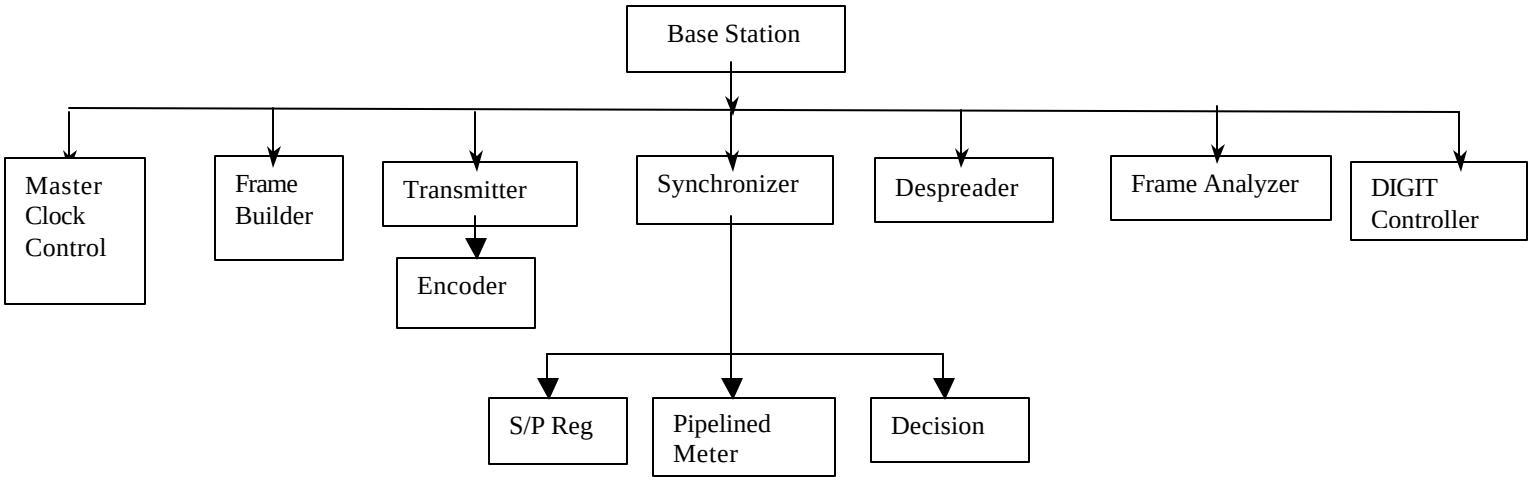
14. David Li, Eric Cheung, Felicia Cheng and Wilson Kwan, “Simple to use LCD driver (DMC 16207)”, EE552 Course Project, winter 2000

## Appendix B – Data Sheets and Schematics

## Appendix C Design Hierarchy

All components are compiled and simulated with no known bugs.



## Appendix D Index to Test Cases

## Appendix E Design Verification

In the following simulation we are testing the Mobile User Board. The transmission is directly fed back into the receive end of the system. We simulate the following:

**Top Level of Mobile User:**

In this simulation we are displaying the entire simulation from 0 μs to 500 μs at a glance in one page. We are expecting the following results:

- Simulation of keypad input and user enabled signals
- Correct encoding of user data
- Construction of the data frame
- Construction of the data frame and channel header.
  The following channel headers are expected:
  0011 – Channel Request
  0111 – Channel Close
  1010 – Communication Send
- Channel Status being either ready or closed
- Transmission of data frame and sync code
- Correct reception of data frame meaning the sync code is detected
- Correct de-spreading of data
- Extraction of data to display on LCD

System Reset:

In this simulation we are testing the system reset on the Mobile User Board for 1 clock period. We are expecting the following:

- All active high signals to go low
- Channel closed to go high
- Keypad row decoder signals to go high (active low)

Channel Initialization for User 1:

In this simulation User 1 goes on-line. We are expecting the following:

- Channel request frame to be built
- Channel request frame to be transmitted
- Channel request frame to be received
- Channel request frame to frame synchronize and be de-spread

Keypad input for User 1:

In this simulation User 1 begins to press some keys on the keypad. We are expecting the following:

- Key-in signal to go high after the simplified de-bounce period of "1111" is reached
- Column outputs to go drive until we find the correct column
- Clock enable signal to go high from the clock master
- Proper encoded result from the keypad

Communication from User 1:

In this simulation User 1 has finished pressing a key and the data is passed on the rest of the components to be relayed to the base station. We are expecting the following:

- To generate the data frame after a key has been pressed
- Correct transmission of the sync code and spread data frame
- Correct reception and de-spreading of the data frame.

Channel Close from User 1

In this simulation User 1 terminates their connection to the system. We are expecting the following:

- Enable signal to go low
- Close frame to be transmitted and received
- Channel ready signal to go low, channel closed signal to go high

These simulations are repeated for User 2 and with both users on-line. However, when both users are on-line we also simulate the following:

User 2 goes off-line and then on-line:

In this simulation User 2 goes off-line while User 1 is still using the channel. Then User 2 goes back on-line. We are expecting the following:

- No close frame to be produced because User 1 is still on-line
- No channel request to be produced because User 1 still has the channel occupied.
- Channel closed signal to stay low, channel ready signal to stay high.

Typical Data Frame:

In this simulation display a typical data communication frame during transmission. We decode the transmitted data by hand and compare it to the actual data frame before spreading and transmission. We expect the following:

- Transmitted spread frame data to match the internal system frame data
- Synchronization code is not de-spread

## Appendix F Index to VHDL Code

Pages have the following format:

| | |
|---|---|
| 1 | 3 |
| 2 | 4 |

( C-compiled, no error;  S-simulated, no known bugs)

## Appendix G VHDL Code

## Appendix H Test Bench Index and Test Bench

In this test bench we are testing for proper behavior of data frame reception and transmission in the base station. We send idle noise into the base station followed by various data frames to see if the base station frame synchronizes to the data frame and if the base station is outputting the correct data frames in response to the input.

In the following test bench we are testing the Base Station board. We simulate the following:

System Reset:

In this test we test the system reset for one clock period. We expect the following:

- All active high signals to go low
- DIGIT display signals to go high (active low)

Build Channel with Idle Frames – User 1:

In this test we send some idle bits followed by a channel request frame from User 1. We expect the following:

- Channel request frame is de-spread and frame synchronized
- Acknowledge frame is created, spread and transmitted out of the Base Station
- No data is to be displayed on the DIGIT display

Send communication frame with idle frames – User 1:

In this test we send some User 1 data to the base station. We send some idle bits followed by two communication frames. In the first frame, we send "010" data for User 1 and void data for User 2. Then we send another frame. In this frame, we send "011" data for User 1 and void data for User 2. We expect the following results:

- Communication frame from Mobile User is de-spread and frame synchronized
- Communication frame from Base Station is built, spread and data is echoed back through the transmitter
- Proper encoding of data in communication frame for DIGIT display output

Send close channel frame – User 1:

In this test, User 1 goes off-line and a close channel frame is sent. We expect the following:

- Close frame from Mobile User is de-spread and frame synchronized
- Empty frame with sync code is create, spread and transmitted back through the transmitter
- No data is to be displayed on the DIGIT display

Build channel with idle frames – User 2:

In this test we send some idle frames followed by a channel request frame from User 2. We expect the following:

- Channel request frame is de-spread and frame synchronized
- Acknowledge frame is created, spread and transmitted out of the Base Station
- No data is to be displayed on the DIGIT display

Send communication frame with idle frames – User 2:

In this test we send some User 1 data to the base station. We send some idle bits followed by one communication frame. In this frame, we send "101" data for User 1 and void data for User. We expect the following results:

- Communication frame from Mobile User is de-spread and frame synchronized
- Communication frame from Base Station is built, spread and data is echoed back through the transmitter
- Proper encoding of data in communication frame for DIGIT display output

Send close channel frame – User 2:

In this test, User 2 goes off-line and a close channel frame is sent. We expect the following:

- Close frame from Mobile User is de-spread and frame synchronized
- Empty frame with sync code is create, spread and transmitted back through the transmitter
- No data is to be displayed on the DIGIT display

Build channel with idle frames – User 1 and 2:

In this test we send some idle frames followed by a channel request frame from the Mobile User Board. We expect the following:

- Channel request frame is de-spread and frame synchronized
- Acknowledge frame is created, spread and transmitted out of the Base Station
- No data is to be displayed on the DIGIT display

Send communication frame with idle frames – User 1 and 2:

In this test we send some User 1 data to the base station. We send some idle bits followed by two communication frames. In the first frame, we send "000" data for User 1 and "111" data for User. Then we send another frame. In this frame, we send "100" data for User 1 and void data for User 2. We expect the following results:

- Communication frame from Mobile User is de-spread and frame synchronized
- Communication frame from Base Station is built, spread and data is echoed back through the transmitter
- Proper encoding of data in communication frame for DIGIT display output

Send close channel frame – User 1 and 2:

In this test, both users go off-line and a close channel frame is sent. We expect the following:

- Close frame from Mobile User is de-spread and frame synchronized
- Empty frame with sync code is create, spread and transmitted back through the transmitter
- No data is to be displayed on the DIGIT display