Apple® Desktop Bus (ADB™)

Author: Rob McCall - WFT Electronics

INTRODUCTION

The purpose of this application note is to introduce a PIC16CXX based ADB interface which can be used as a basis for the development of custom ADB devices. This application note describes the hardware involved, a general purpose ADB protocol handler and an example application task. The example software application supports a single key keyboard to the Macintosh® computer (see Figure 1).

OVERVIEW

ADB licensing from Apple Computer.

Described as a peripheral bus used on almost all Macintoshs (except for the Macintosh 128, 512k, and Plus) for keyboards, mice, etc.

Communication between the ADB task and the application task takes place using several flags. The flags indicate whether there is data received that needs to be sent to the Macintosh, or if data from the Macintosh needs to be sent by the application.

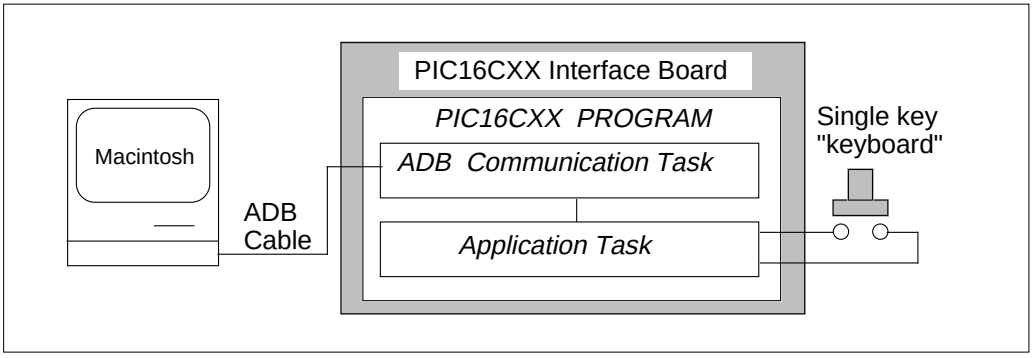
EXPLANATION OF ADB TECHNOLOGY

The ADB is an asynchronous pulse-width communication protocol supporting a limited number of devices. All devices share a single I/O wire in a multi-drop master/slave configuration in which any slave devices may request service. This accomplished through a wired OR negative logic arrangement.

The ADB cable is composed of four wires: +5v, gnd, ADB signal, and power-on (of the Macintosh). The signal wire communicates ADB input and output using an open collector type signal. The number of devices is limited by the addressing scheme and a maximum current draw of 500mA.

Every ADB device has a default address at start-up assigned by Apple. If there are device address conflicts, the protocol supports the reassignment of device addresses at start-up. The software in the PIC16CXX discussed here is designed to easily modify the device address to make the PIC16CXX appear as another ADB device for testing and development.

FIGURE 1: BLOCK DIAGRAM OF FUNCTIONALITY



No device issues commands, except the host. However, devices are permitted to request service during specific time intervals in the signal/Command protocol. A Service Request is referred to as an "**Srq**" The signal protocol communication is accomplished by pulling the ADB line low for various time intervals.

The host controls the flow of data through issuance of specific signal sequences and by issuing several types of Commands. The basic Command types are Talk, Listen, Flush, or Reserved. Each Command has a component called a "Register" indicator which specifies the storage area affected by the Command type. The following is a summary explanation of the each of the Commands. The complete specifications are available from Apple, as listed in the Resources section.

PROTOCOL ASSUMPTIONS

The ADB protocol is defined with a number of general assumptions about its use. These assumptions have driven the general philosophy of the communication sequences. It is assumed that the devices on the ADB are used for human input and each are used one at a time, such as a keyboard and a mouse. It is also assumed that the transfer time from one device to another is relatively slow. This does not mean that the protocol is limited to these assumptions but rather that the protocol is optimized towards this type of use. This is made very evident in the host polling logic, where the host continues to poll the last device communicated with until another device issues an Srq. Consequently, if another device issues an Srq, the device being communication with (or the host) may need to retransmit.

ADB Elements:

The ADB protocol has two components, a signal protocol and a Command/data protocol. These two elements are intertwined. The signal protocol is differentiated in most cases by timing periods during which the ADB signal is low. The Apple ADB specification allows +/- 3% tolerance timing of the signals from the host and +/- 30% by the devices. The signals are:

- Reset: signal low for 3 ms.
- Attention: signal low for 800 μs.
- Sync: signal high for 70 μs.
- Stop-to-Start-Time (**Tlt**): signal high for 160 to 240 μs.
- Service Request (**Srq**): signal low for 300 μs.

After device initialization, in general all communication through the ADB is accomplished through the following event sequence initiated by the host: **Attention** Signal, **Sync** Signal, **Command Packet**, **Tlt** signal, and **Data Packet** transfer. Depending upon the Command, the device may or may not respond with a data packet. Service Requests are issued by the devices during a very specific time at the end the reception of the Command packet.

The Command packets and the Data packets are the constructs used to communicate the digital information. The method of representing data bits is accomplished in a signal timing construct called a **bit cell**. Each **bit cell** is a 100 μs period. Data 1's and 0's are defined by the proportions of the bit cell time period when the line is low and then high. A "1" bit is represented by the line low for 35 μs, and high for 65 μs. Conversely, A "0" bit is represented by the line low for 35 μs, and high for 65 μs (see Figure 3).

FIGURE 2: TYPICAL TRANSACTION WITH COMMAND AND DATA

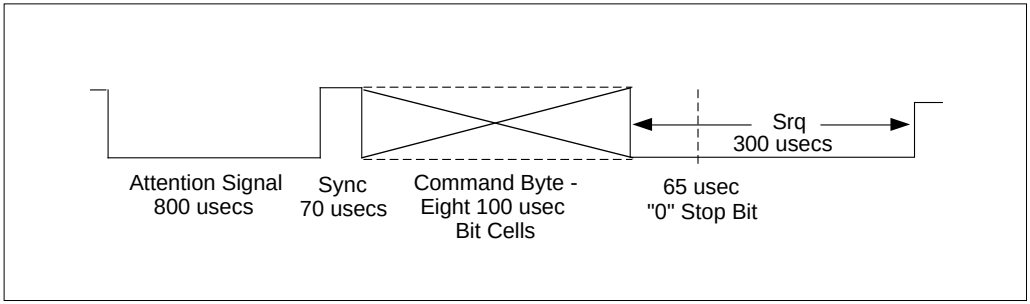
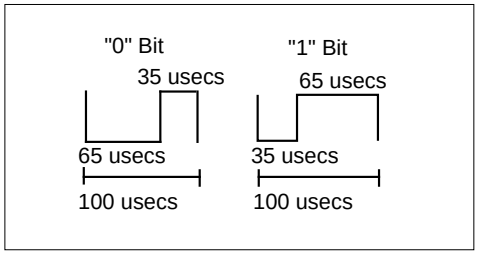


FIGURE 3: BIT CELLS



The **Command Packet**, received from the host, follows an Attention signal and a Sync signal. It consists of an 8 bit Command Byte and a "0" Command Stop Bit. The Command Byte may be broken down into two nibbles. The upper nibble is a four bit unique Device Address. The lower nibble is defined as a Global or Reserved Command for all devices, or a Talk, Listen, or Flush Command for a specific device. Also contained in the lower nibble is a "Register" designator which further details the Command. The importance of the Command Stop bit cell is that **Srq**'s can only be issued by a device to the host during the Command Stop Bit cell low time if the device address is not for the device wishing service, The Host controls when Srq's are allowed through the Command protocol. The Tlt signal and Data Packet transfer, which are part of every Command packet signal sequence, are overridden if an Srq is issued by any device.

A **Data Packet** is the data sent to, or received from, the host. Its length is variable from 2 to 8 bytes. The structure is a "1" Start Bit, followed by 2 to 8 bytes, ending with a "0" Stop Bit. The Apple ADB documentation refers to the data packet sent or requested as Device Data "Registers". This does not necessarily indicate a specific place in memory. In this PIC16CXX implementation, each Data Register has been limited to two PIC16CXX register bytes. The ADB specification allows each Data Register to hold between two and eight bytes. They are referenced in the Command byte as "register" as 0, 1, 2, or 3. Data Register 3 has special significance. It holds the special status information bits (such as whether Srq's are allowed), the Device Address, and the Device Handler ID. Commands are further defined by the "register id" sent in the Command data packet.

For example, if the Host issues the Command in binary of 0010 1100, it would be interpreted as "Device 2, Talk Register 0". The complete definition of the Commands and data registers are described in detail in the ADB specifications supplied by Apple.

PIC16CXX ADB PROTOCOL PROGRAM EVENT SEQUENCE:

Overview

At power-on the host will generate a Reset signal. The purpose of Reset is to initialize the devices on the ADB line. This includes determining the addresses of each device, and resolving device address conflicts if there are any. Once the device addresses are determined, each device waits to be Commanded or issues an Srq if it requires service from the host and is not being addressed by the host. After Reset processing the ADB Protocol Task monitors the ADB line for the Attention/Sync/Command signal sequence. The PIC16CXX program differentiates the signal timing.

Note: The signal detection routines check to see if the Application Task needs service after each event and after the falling edge of the Attention signal is detected.

Command interpretation is accomplished during the low signal time of the Stop Bit cell of the Command packet. Response to the Command must occur after the minimum time of the Stop to Start time period (**Tlt**), which is 160 usec. but before the max Tlt time of 240 usecs. When a device has issued an Srq, it waits to be addressed by the host. If the next Command received is not for the device, it issues the Srq again. The normal response to an Srq will be a Talk Command from the host.

Detailed Description

START-UP

Upon start-up, the Reset routine is executed, looking for the ADB line to be high. When the line is high, an initialization routine is executed during which registers are cleared or loaded with default values. The only exception is a register for generating a random address used in the address conflict resolution process.

RESET

During a Reset condition, default values are loaded, such as the Default Device Address and Handler ID (a piece of information used by the host to identify the type of device). More than one device may have the same address. There is a sequence of events to resolve address conflicts described in the Implementation section. The host assigns a unique address to each device. The Reset condition only takes place once, during start-up, except under unusual conditions, such as testing this program.

ATTENTION ROUTINE

When the Reset routine is complete, the Attention Signal routine is executed, looking for the line to go low and then high. This low time is monitored to be within range of the Attention Signal Timing. If the timing is below the minimum threshold, the routine aborts to start over again looking for the line to go low as the beginning of the Attention Signal. If the low time is exceeded, the routine aborts to the Reset Signal routine.

SYNC SIGNAL ROUTINE

When the line transitions to high, the Sync Signal routine looks for the line to go low as the start of the first bit of the Command Byte. If the Sync high time is exceeded, the routine aborts to the Attention Signal.

COMMAND ROUTINE

The Command routine detects and decodes the next 8 bit cells as the Command Byte. The routine must first determine if the device address is for the device. If the routine determines that the device address in the Command matches the stored device address it may do one of two things; issue an Srq to the host by holding the line low, or go on to check if the Command is Global to all devices. If Global, the routine determines the specific Command and executes the routine for that Global Command. After execution of the Command routine it then goes back to look for the Attention Signal.

When a device is addressed, it determines whether the Command is to Talk, Listen, or Flush data, for the specified Data Register number. If the Command is for Data Register 3, there are special considerations, described for this program in the Implementation section below. If the Command is to Flush, the routine clears the data in the specified Register. The ADB specification defines the action of the Flush Command to be device specific. For a Talk Command or Listen Command, the device then waits for the Tlt signal. When the Command is to Talk, the device sends the data bytes from the specified register and a Data Stop Bit after the Tlt minimum time. For a Listen Command, the device receives data for the specified Register.

When the data has been Flushed, Sent, or Received, the program the device then goes to look for the Attention signal again.

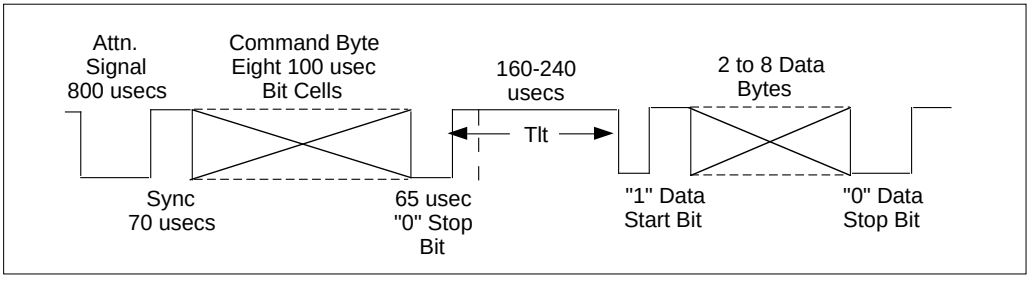
Notes: 1. In this PIC16CXX program, the Application Task is serviced before looking for the Attention signal.

2. If at any time the line is low or high outside of timing ranges, the program aborts to check if an Attention or Reset signal has been issued by the Host. In the case of sending Data, the program goes first to the Collision routine.

SENDING DATA TO THE HOST

Data is sent only in response to a Talk Command. For every data bit cell, the line is tested to go high at the proper time. If the line is still low, a collision has occurred. When a collision is detected, a collision flag is set, and the program aborts to look for a Command signal sequence.

FIGURE 4: TYPICAL TRANSACTION WITH SERVICE REQUEST



IMPLEMENTATION

Hardware:

The hardware of this circuit is fairly simple. The circuit is powered via the +5v and Gnd wires of the ADB cable. The ADB I/O wire is connected to port RA0 with a pull-up resistor to 5v. The T0CKI pin is tied to Gnd. The Master Clear pin is tied to 5v.

This circuit uses a 4Mhz crystal as a timing reference, but higher values may be substituted. The software is designed to accommodate higher frequencies.

A push-button switch is used as the single key of the "keyboard". One side is connected to port RB1 with a pull-up resistor to 5v, and the other side to Gnd. An LED is used to indicate that the 'key' has been pressed, with the positive side connected to port RB0 and the negative side to Gnd.

Software:

There are two sections of the program designated as "Application Tasks", setup to switch between a protocol support task for the ADB signal decode and processing, and another task, the Application Task, in this case a single key "keyboard" routine. The ADB protocol task has priority. The first section of the code is the ADB protocol task, the second section is the Application task, "Keyboard". The two tasks communicate through flags which indicate that data needs to be sent, or that data has been received.

The Keyboard Task is run at two times; 1) during the Attention Signal, 2) between the end of the Data Stop Bit and the beginning of the Attention Signal. The Keyboard Tasks is given up to 500 usecs during the Attention Signal, and 900 usecs during the time between the end of the Data Stop Bit and the beginning of the Attention Signal. It is important to note here that the other tasks MUST NOT AFFECT TMR0 or the ADB time variable that the Attention Signal is using to keep track of the RTCC.

Timing:

Timing of is accomplished by first loading a constant into a time variable. This constant represents the maximum limit for the current routine, which may not necessarily be the maximum timing range for the current Signal. The TMR0 value is loaded into the working register, and subtracted from the time variable. The Carry bit of the Status register is tested to see if it is set or cleared. If the bit is clear, the current timing limit has been exceeded. Further action is taken based on this status. It is important that the constant not be allowed very close to 255, or rollover may occur, giving inaccurate results. The prescaler is applied to the TMR0 as necessary.

The following are the timing ranges used by this program for ADB signals:

Reset	Greater Than 824 usecs
Attention	776-824 usecs
Sync	72 usecs
Bit Cell	Up to 104 usecs
1 Bit low time	< 50 usecs
0 Bit low time	> 50 < 72 usecs
Stop bit	0 Bit
Stop to Start (Tlt)	140-260 usecs
Service Request (Srq)	300 usecs

Note: The range of values given for 0 bit, 1 bit, and Tlt timing are slightly wider than those given in the ADB specification.

How Address Conflicts are Resolved:

During the start-up process the host sends a "Talk Register 3" Command to each device address, and waits for a response. When a device recognizes that the Host issued a "Talk Register 3" Command, it responds by sending a random address. During the transfer of each bit cell of the random address the signal line is monitored for the expected signal level. If the signal is not what is expected there is an address conflict. If the address is sent successfully the host will respond with a Listen Command to that device with a new Device Address for that device to move to. The device then only responds to Commands at the new Address.

If there is a conflict, where two devices have the same default address, and respond at the same time, the device that finds the line low when it expects it to be high, immediately stops transmitting because it has determined that a collision has occurred. The device which detected the collision marks its address as unmovable and therefore ignores the address move Command, a Listen register 3 Command, The device maintains the unmovable address condition until it has executed a successful response to the talk register 3 Command.

The host continues sending a Talk Register 3 Command at the same address until there is a time-out and no device responds. This is how conflicts are resolved when more than one device has the same address; for example, if two keyboards are connected.

Program Sequence:

Words in parenthesis accompanying the TITLES are Labels of procedures corresponding in the code.

Start-up / IDLE (Start)

Start by setting the ADB pin on Port A and the Switch Pin on Port B as inputs, and tri-stating the rest of Port A and B as outputs.

INITIALIZE DEFAULT VALUES WHEN THE LINE IS HIGH (Reset)

Look for the line to be high, and when it is, clear or initialize registers to default values.

LOOK FOR ATTENTION OR RESET (AttnSig)

Look for the line to go low, when it does, clear TMR0 and time how long it is low. An Attention Signal has occurred when the line goes high between 776 and 824 usecs. If the low time is measured less than 776 usecs, another signal has occurred and the program aborts, looking for the Attention Signal again. When the low time is measured greater than 824 usecs, the program interprets this timing as a Reset Signal. The program starts over again, waiting for the line to be high, and when it is, performs a Reset initialization.

Note: The keyboard task is performed during the attention signal (Task_2).

LOOK FOR SYNC SIGNAL (SyncSig; calls Srq)

The Sync Signal is the high time between the rising edge of the Attention Signal and the falling edge of the first bit of the Command.

GET THE COMMAND (Command; calls Get_Bit)

Look for the Command; a combination of eight 0 and 1 bits. The MSB is sent first. This is achieved by calling a the Get_Bit routine, which checks whether the maximum Bit Cell time is exceeded, if not, it looks for the rising edge at the end of the bit. When the bit is received, it is rotated into a variable, and the end of the bit cell is expected. When the falling edge of the next bit is detected, the routine clears TMR0 and returns to Command, which calls Get_Bit again until all 8-bits of the Command have been received.

ISSUE A SERVICE REQUEST IF NECESSARY (Srq)

If data needs to be sent to the Host, a Service Request (Srq) is issued by holding the line low while the Stop Bit is being received, during the Stop-to-Start time (Tlt) between the end of the Command Stop bit and the beginning of the Data Start Bit

LOOK FOR STOP BIT (CmdStop)

Look for the Stop Bit (a 0 bit of 65 usecs) that comes after the last Command Byte.

INTERPRET THE COMMAND (AddrChk)

After the Command has been received, determine if the Address belongs to this Device. If the Address is not for this Device, determine if the Command is global for all Devices and if so, do that Command. If this is not a Global/Reserved Command, call the Service Request (Srq) routine to see if an Srq should to be issued to the Host, and do so if necessary, then return to get the Attn Signal. If the Address is for this Device determine whether it is a Talk, Listen, or Flush Command, and go to the specified Command routine.

SENDING DATA (Talk; calls Tlt)

If the Command was interpreted to be a Talk Command addressed to this Device, call the Stop-to-Start Time (Tlt) routine. When the Tlt routine has completed, determine if this is a Talk Register 3 Command. If so, return a Random Address as part of the two bytes sent to the Host. If this is not a Talk Register 3 Command, determine if Data needs to be sent. If so, send the Data Start Bit (a '1'), two bytes of Data from the indicated register, and a Stop Bit (a '0'). If not, abort to the Attention Signal. If at any time the transmission of Data is interrupted, abort to the Collision routine. Only after a complete transmission should the flags be cleared indicating a successful transmission.

Note: The ADB Specification indicates data may be between two and eight bytes long. The limitations of the PIC16C54/55/56 parts allow only two bytes of data to be sent by this program due to limited register space. If more than two bytes of data must be sent, use the PIC16C57.

RECEIVING DATA (Listen; calls Tlt)

If the Command was interpreted to be a Listen Command addressed to this Device, call the Stop-to-Start Time (Tlt) routine. When the Tlt routine has completed, receive the rest of the Data Start Bit, 2 Data Bytes, and Data Stop Bit. When the Data has been received, determine whether this is a Listen Register 3 Command. If this is a Listen Register 3 Command, interpret what the Command is. If this is a conditional Address change Command, determine if this Device's Address is moveable at this time. If not, abort to the Attention Signal. If so, change the Device to the new Address and go run the Second Application Task. If this is not a Listen Register 3 Command, move the Data into the specified register and go run the Second Application Task.

LOOK FOR THE STOP TO START TIME (Tlt)

After the Command and Stop Bit, the Talk or Listen routines call the Tlt routine. Tlt looks for the line to go low. If the line went low before the Min. Tlt Time, see if this is a Talk Command. If this is a Talk Command, abort to the Collision routine. If this is a Listen Command, abort to the Attention Signal.

If the Min. Tlt time passes and the line is high, see if the Talk routine called the Tlt, if so, go wait for until the middle of the Tlt, then return to the Talk routine to send the Data Start Bit, Data Bytes, and Stop Bit. If at any time the line goes low during the Tlt and the Talk routine called it, abort to the Collision routine.

If the Listen routine called Tlt, look for the line to go low as the beginning of the Data Start Bit. When the line goes low, return for the rest of the Start Bit. If the line doesn't go low before the Max. Tlt time is up, abort to the Attention Signal.

THE KEYBOARD TASK IS PERFORMED BETWEEN THE END OF THE DATA STOP BIT AND THE ATTENTION SIGNAL (Task_2)

The Keyboard Task checks to see if the key has been pressed. When the key is pressed, flags are set to indicate this and an LED is turned on until the key has been debounced. The flags allow the key to be debounced, Srq(s) to be sent to the Host, and indicate to the Talk routine that Data needs to be sent. Two bytes of data are loaded into Register 0 representing a key-down code and a flag is set indicating to the ADB task that data needs be sent to the host. When the key-down codes have been sent, the key up codes are loaded into Register 0. When the key-up codes have been sent and the key has been debounced, the flags are cleared. The final routine of Task_2 decides whether to return to the beginning or middle of the Attention Signal.

FIGURE 5: APPLE DESKTOP BUS PIC16CXX FLOWCHART

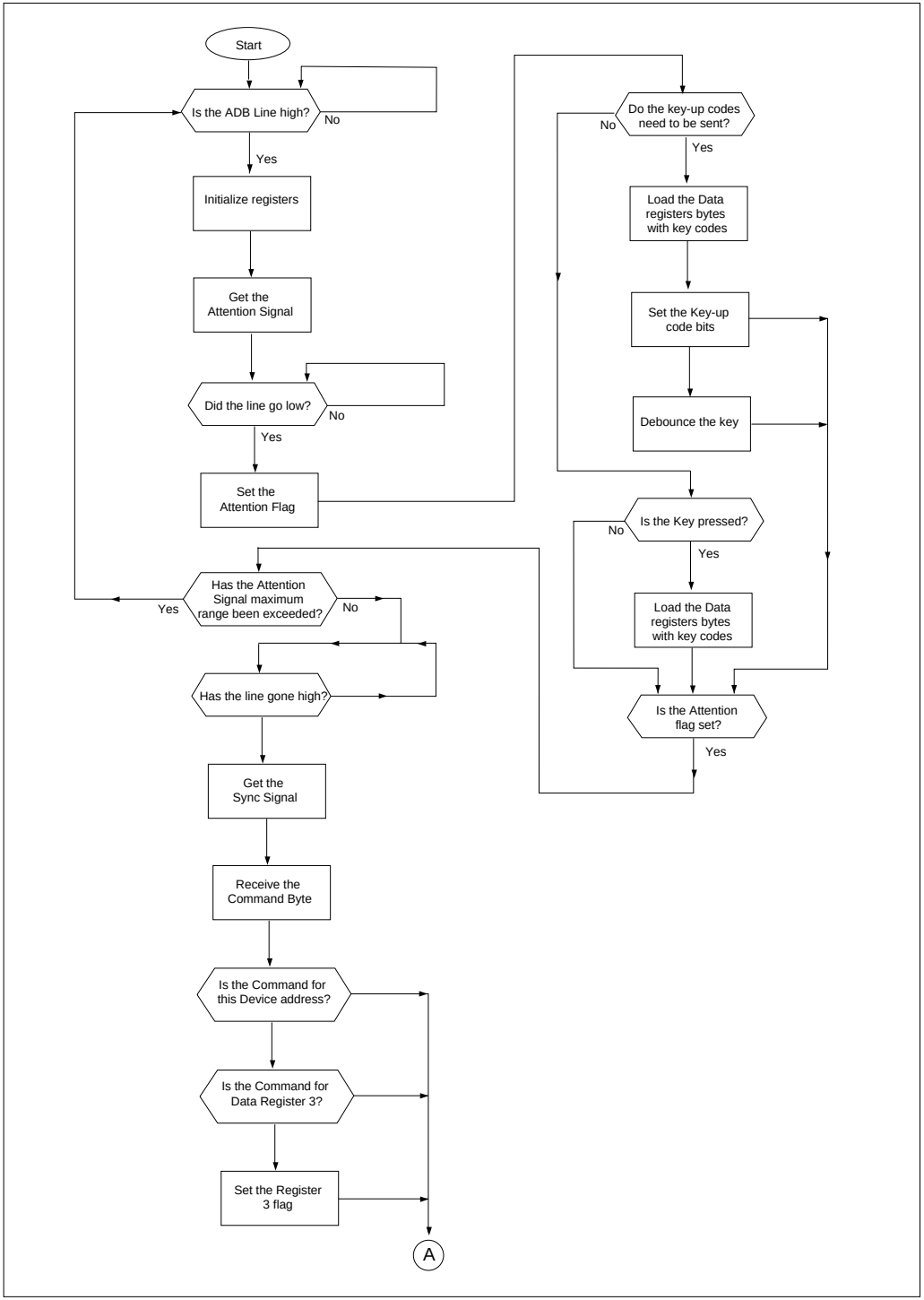
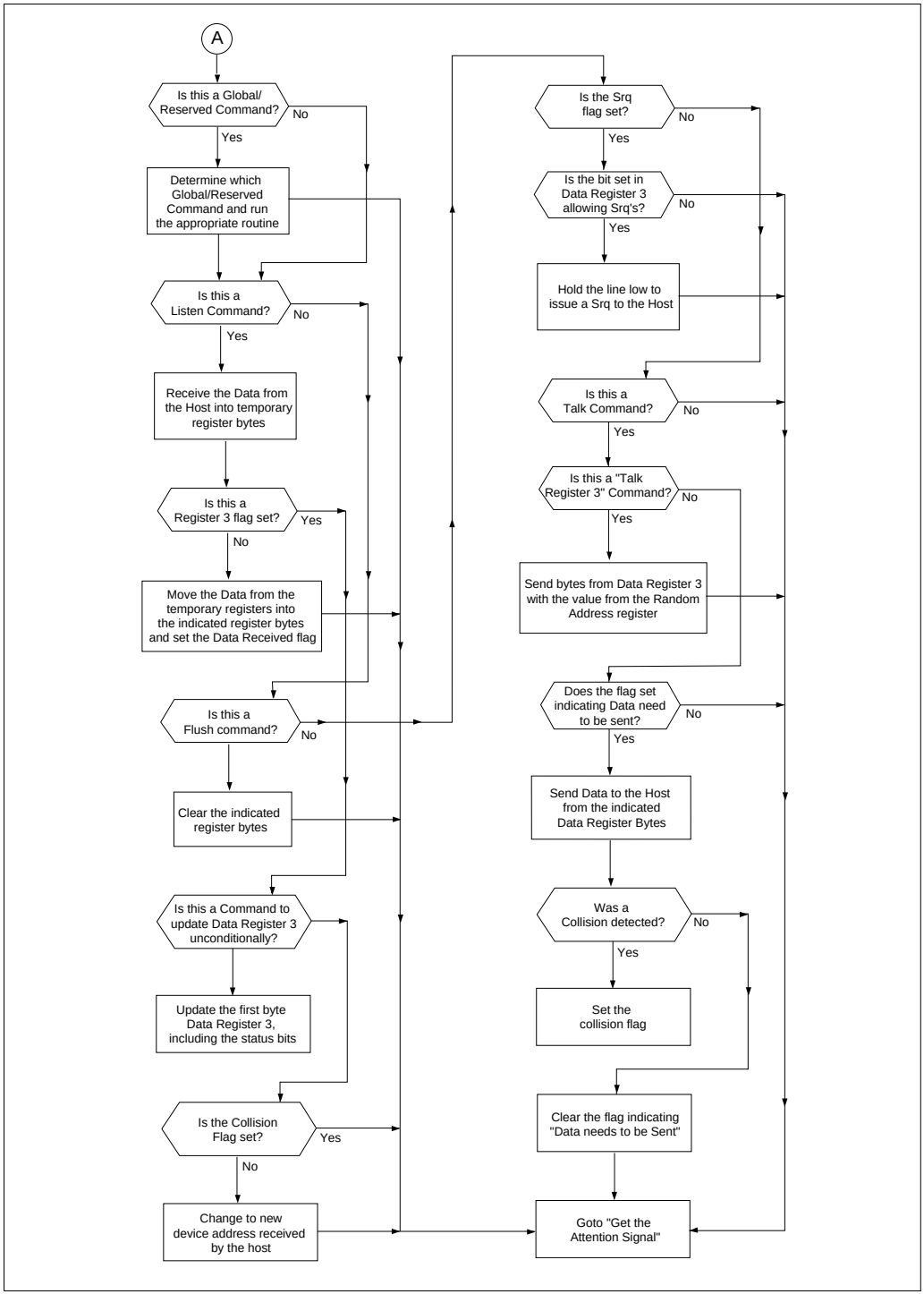


FIGURE 5 (CONT.): APPLE DESKTOP BUS PIC16CXX FLOWCHART

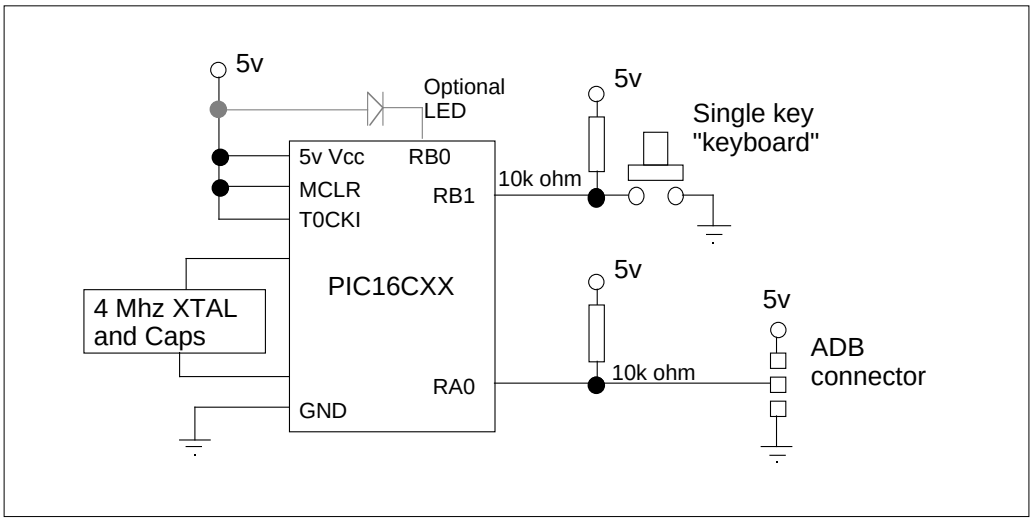


SUGGESTIONS ABOUT MODIFYING THE CODE

1) If high crystal frequencies are used, a divider equate at the beginning of the timing section of the equates allows an easy adaptation for all established timing definitions.

2) The second application task may occur as a communication task with another PIC16CXX chip by using the three other i/o lines on Port A, although code for this has not yet been written to test this. Two of the lines would be used as ready-to-send (one for each PIC16CXX). The third would be used as a data line, using low signals as 0 bits, and high signals as 1 bits. Additionally, all eight lines on Port B may be used as well.

FIGURE 6: SIMPLE SCHEMATIC OF THE TEST BOARD



RESOURCES

Apple Publications and Support Software

MacTech Magazine (formerly MacTutor) is a publication dedicated to supporting the Macintosh. They have had several articles regarding the Apple Desktop Bus. They publish a CD-ROM that contains all of their articles from 1984 to 1992. Also, single disks are available (ask for #42).

MacTech Magazine can be contacted at:

P.O. Box 250055
Los Angeles, CA 90025-9555
310 575-4343 FAX 310 575-0925
Applelink: MACTECHMAG
Internet: info@xplain.com

Apple licenses the ADB technology. They can be contacted at:

20525 Mariani Ave.
Cupertino, CA 95014
Attn: Software Licensing

- Apple Keyboard, extended, specification drawing #062-0168-A.
- Apple Desktop specification drawing # 062-0267-E.
- Apple Desktop connector, plug, Mini DIN drawing #519-032X-A.
- Engineering Specification, Macintosh transceiver interface, ADB drawing #062-2012-A.
- Apple keyboard, specification drawing #062-0169-A.
- Developer CD series, Tool Chest Edition, August 1993 contains:
    - Folder = Tool Chest: Devices and Hardware: Apple Desktop Bus
    - ADB Analyzer
    - ADB Parser (most complete environment)
    - ADB Lister
    - ADB ReInit
    - ADB Tablet code samples

WFT Electronics offers free assistance in procuring necessary ADB info. Contact Gus Calabrese, Rob McCall, Dave Evink at:

4555 E. 16th Ave.
Denver, CO 80220
303 321-1119 FAX 303-321-1119 Applelink: WFT
Internet: Gus_Calabrese@onenet-bbs.orgA

AUTHOR / CREDITS

Rob McCall developed the majority of the PIC16CXX ADB code. He also wrote most of the application note. Gus Calabrese, Dave Evink, and Curt Apperson supported this effort. Dave works with Gus, Rob, and Curt in developing a variety of embedded processor products.

Contact Gus Calabrese, Rob McCall, Dave Evink, Curt Apperson at:

WFT Electronics
4555 E. 16th Ave.
Denver, CO 80220
303 321-1119 FAX 303-321-1119 Applelink: WFT
Internet: Gus_Calabrese@onenet-bbs.org

NOTES:

WORLDWIDE SALES & SERVICE

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602 786-7200 Fax: 602 786-7277
Technical Support: 602 786-7627
Web: http://www.mchip.com/microhip

Atlanta
Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770 640-0034 Fax: 770 640-0307

Boston
Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508 480-9990 Fax: 508 480-8575

Chicago
Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 708 285-0071 Fax: 708 285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton
Microchip Technology Inc.
35 Rockridge Road
Englewood, OH 45322
Tel: 513 832-2543 Fax: 513 832-2841

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 455
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516 273-5305 Fax: 516 273-5335

AMERICAS (continued)

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408 436-7950 Fax: 408 436-7955

ASIA/PACIFIC

Hong Kong
Microchip Technology
Unit No. 3002-3004, Tower 1
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

Korea
Microchip Technology
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku,
Seoul, Korea
Tel: 82 2 554 7200 Fax: 82 2 558 5934

Singapore
Microchip Technology
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65 334 8870 Fax: 65 334 8850

Taiwan
Microchip Technology
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886 2 717 7175 Fax: 886 2 545 0139

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44 0 1628 851077 Fax: 44 0 1628 850259

France
Arizona Microchip Technology SARL
2 Rue du Buisson aux Fraises
91300 Massy - France
Tel: 33 1 69 53 63 20 Fax: 33 1 69 30 90 79

Germany
Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 Muenchen, Germany
Tel: 49 89 627 144 0 Fax: 49 89 627 144 44

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 039 689 9939 Fax: 39 039 689 9883

JAPAN

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81 45 471 6166 Fax: 81 45 471 6122

9/22/95