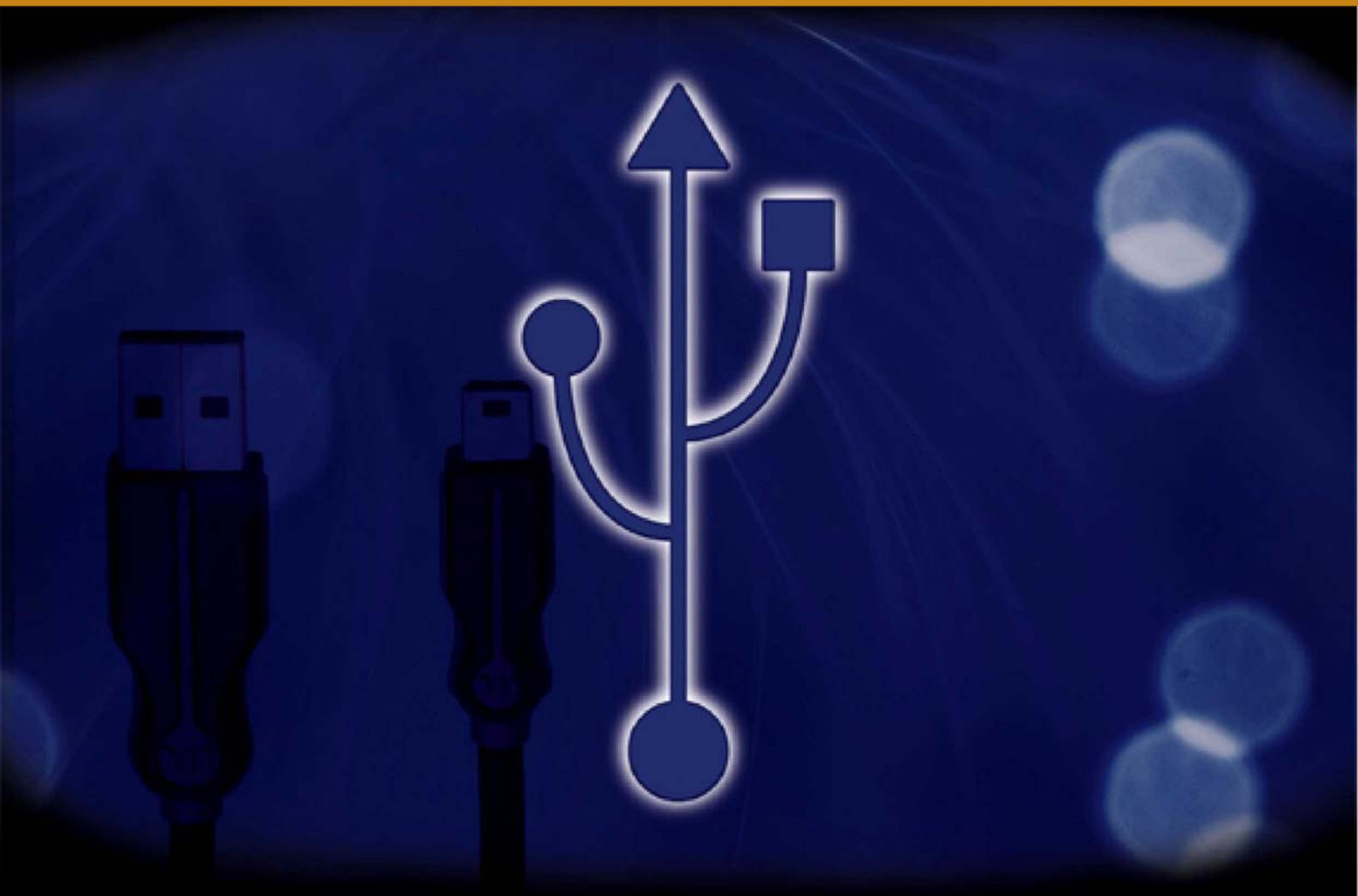
eBLOCKS®

LEARN·DESIGN·BUILD

USB Systems



EB540-80-02

MATRIX

www.matrixmultimedia.com

EB540

USB

Course notes

About this course

Aims: The aim is to introduce the concepts involved in USB devices.

On completing this course students will have learned about:

- the relationship between USB masters, hubs and endpoints;
- the electrical principles behind USB architecture;
- the components that make up a USB device;
- the options available for USB devices;
- the addressing schemes;
- USB signals and routing;
- low power and sleep modes;
- USB device drivers;
- USB devices that do not require drivers.

What you will need:

To complete this course, students will need the following equipment:

- Flowcode software
- E-blocks including:
  - 1 Multiprogrammer (PIC - EB006)
    - with PIC18F4455 device and 4MHz crystal;
  - 1 Sensor E-Block (EB003);
  - 1 LED E-Block (EB004);
  - 1 LCD E-Block (EB005);
  - 1 Keypad E-Block (EB014);
  - 1 USB E-Block (EB055);
  - 1 RS232 E-Block (EB015);

Using this course:

This course presents students with a number of tasks listed in the exercises that follow the USB overview. All the information needed to complete these is contained in the notes.

Before starting the exercises, students should familiarise themselves with the background material.

Time:
To undertake all of the exercises will take around twelve hours.

Important note:
Information presented here is correct at the time of publication..
Please check the Matrix Multimedia web site, http://www.matrixmultimedia.com for the latest E-Blocks documentation.

Scheme of Work

Section	Notes for instructors	Timing (minutes)
1. Introduction to USB		
1.1 Preamble	Students familiarise themselves with the course ahead. They can use a web browser to review the differences between various versions of USB, from USB1.0 to USB 3.0.	5 -10
1.2 Key Advantages of USB	This section lists the main advantages, and drawbacks, of the USB protocol. Again, they can use the internet to familiarise themselves with other communication protocols such as Firewire and RS232.	5 - 15
1.3 Introduction to USB	This section compares the performance of USB 2.0 devices with Firewire and Serial communications links. It then gives a glossary of terms used in USB systems, including: master / slave configuration; USB power; connectors; functions; endpoints; pipes; classes device drivers addressing enumeration interface speeds noise immunity. USB is a network of attachments connected to a host computer. The attachments are either functions or hubs, and together they are known as devices. The host has a hub embedded in it called the root hub, the interface between the computer and the USB ports it houses. External devices which offer more than one function are usually combined with a hub, and are then called compound devices. Hubs may be connected to other hubs in a tiered arrangement, but logically the system appears as a linear bus. These terms may mean little until the student has had hands-on experience of the USB protocol later in the course. However, it pays to spend time on these terms now. At least students will know where to find some explanation of these terms if problems arise. No attempt has been made to tackle physical layer issues, such as the use of NRZI for signalling, and 'bit stuffing'. The terms 'J state' 'K state' and 'Single-ended zero (SE0)' are not used. Instructors wishing to expand on these issues will find the solution a fitting tool to facilitate this.	20 - 30

2. Transfer Types		
2.1 Transfer Types	The USB protocol is used across an increasingly wide variety of applications. Different situations demand different kinds of data transfer. This section outlines the differences between control, interrupt, bulk and isochronous transfers. Depending on the situation, the designer can choose to prioritise data validity, latency (delay) or bandwidth by choosing the appropriate transfer type. Once again, the students can use the internet to reinforce the ideas introduced here, or the instructor may choose to spend time supporting them. Students should be warned that, despite the name, interrupt transfers do not cause interrupts. The text explains that this transfer type is used where previously devices would use interrupts to initiate communication.	15 - 30
2.2 Transfers and Transactions	It is important that students understand and use the correct terminology in USB systems. This section distinguishes between data transfers and transactions. A data transfer may be split across several individual transactions. These transactions may occur across a number of frames. The reality is that the host will send out frames every millisecond. These frames will contain a number of transactions, sandwiched into time slots within the frame. Several transactions within a frame may be addressed to the same device. Each transaction is made up from a number of packets, known as token, data and handshake packets. Some sources refer to these as phases. Each has its own function, and as a result, contains different sets of fields. Students should study the diagrams carefully, so that they understand them. It may be profitable for them to make copies of them for their records.	15 - 30
2.3 Transactions	This section examines the four transaction types – start of frame, token, data and handshake. The purpose of each is described briefly, and their structure is outlined diagrammatically. As its name suggests, the Start-of-frame packet is found at the beginning of every frame. In other words, the host produces one of these every millisecond. One job of the Start-of-frame packet is to track frame number. One of the fields, the frame number, identifies each frame of the transaction. Devices can use this to confirm which transaction has been received, or can use it as a timing source. As it is eleven bits long, it can cope with 2^{11} (= 2048) frames. When the maximum is reached, the frame count resets.	20 - 30

2.3 Transactions continued...	The frame can contain eight microframes, when working at high speed. All eight carry the same frame number. The text goes on to outline the three types of data packet – Setup, IN and OUT. The communication 'pipes' that are set up between the host and peripheral devices are uni-directional. One task of a data packet is to define the direction of data flow. This is always taken from the viewpoint of the host. IN means flowing in to the host and so out from the peripheral. OUT means flowing out of the host, and so in to the peripheral. One implication of this, explored later, is that 'Set' requests, where the host imposes a configuration value on the peripheral device, have direction OUT, whereas 'Get' transactions, where the host requests settings from the peripheral device, have direction IN. The Setup process has its own section later. The idea of having two varieties of data packet, called Data0 and Data1, offers another form of error checking, where data is transmitted using multiple transactions. The first of these will use a Data0 packet, the second a Data 1, then a Data0, and so on. The data toggle value is specified in the PID. Both transmitting and receiving devices can monitor the data type to check for missing transactions. The receiver of data will reply with a handshake packet to indicate the status of the transfer. Hence, for an OUT transfer, the peripheral device replies by sending the handshake packet, whereas for IN communications, the host replies. Peripheral devices can reply with ACK (valid data was received,) NAK (the device is busy and did not receive the data,) or STALL (the device does not understand the transfer, or is not active.) A host can only send ACKs. If the receiver detects an error, it returns no handshake packet.	
2.4 USB packets	USB transmissions are synchronous, thanks to NRZI encoding and bit stuffing, which allow the receiver to synchronise its clock with that of the transmitter. In addition, each packet starts with a synchronising field, a series of alternating bits, to ensure that the clocks in the host and peripheral device are in synchronisation. A PID field follows. The term 'PID' has two possible meanings in the USB world. Here, it means Packet IDentifier. When referring to whole devices, it can mean Product Identifier, a 16-bit number used to identify the appropriate device driver. The Packet ID is used to identify the type of packet being sent. e.g. token, data, handshake etc. The table shows how it does this. The student text says that the four most-significant bits are the inverse of the four least significant bits. To be precise, they are the 1's complement of the fout least-significant bits. It is left to the instructor to decide whether to expand on this with the students.	20 - 30

2.4 USB packets continued...	The address field contains a seven bit address, allowing 2^7 (=128) addresses. Only 127 of these are assigned as device addresses. Address 0 is reserved for the mandatory default endpoint on all devices,, so that the host can send control transfers. The endpoint field identifies the endpoint (function) to which the packet is directed. Each endpoint has a number from 1 to 15, expressed as a four-bit binary number. The CRC (cyclic redundancy check) is included to check the data for errors. A mathematical operation is applied to the data at the transmitting device, and the result of that operation is sent as part of the transfer. The same mathematical operation is applied to the data at the receiver. The result is compared to that sent in the CRC. If they are the same, there is no error. If the results are different, then an error is present. The end-of-packet field indicates that the transaction is complete. The bus then goes back to its idle state.	

3 Setup		
3.1 The Setup stage	The purpose of the Setup stage is explained, as the process by which the host learns about the recently attached device, and then each of the three phases, token, data and handshake are described. The core of the transaction is the request for information. The data phase contains five fields, which occupy eight bytes. The first, bmRequestType, specifies the type of request, its direction and the recipient's address. The next, bRequest, specifies the actual request. Its contents depend on the type of request (standard, class, or vendor) identified in the bmRequestType field. The next field, wValue, occupies two bytes and contains information for the recipient from the host. The significance of the information depends on the type of request. For example, the Set_Address request will send the new address in the wValue field. Next comes another two byte field, called wIndex. Again it is used by the host to pass information to the device. This information again depends on the request. It may include an endpoint address, and interface number etc. Finally, comes another two byte field, wLength, The host will specify how many data bytes are sent. The manual then includes a series of tables describing the structure of a number of different requests. The purpose is to illustrate what goes into these requests, and how they are transmitted. It is reference material to aid later study. It is not intended that the students should in any way 'learn' these tables. The handshake phase takes place if the device receives the full transaction without detecting any errors.	20 - 30

4	Learning about USB Device Capabilities		
4.1 4.2 4.3 4.4 4.5 4.6	Descriptors USB Device Descriptors USB Configuration Descriptors USB Interface Descriptors USB Endpoint Descriptors USB String Descriptors	During enumeration, the host learns about the capabilities offered by the device, using control transfers which request a series of descriptors. These start with the broad brush strokes of the Device Descriptor, which cover the global properties of the device. It also specifies all of the subordinate descriptors needed by the host. This is followed by one of the Configuration Descriptors, which include its power requirements, the Interface descriptors, providing information about a feature of the device, including class and protocol information, the endpoint descriptors, which specifies the maximum packet size the endpoint is capable of handling, and finally any optional descriptors. Again, the purpose is illustrative and to act as a reference. It is not intended that the students should in any way ‘learn’ these tables.	20 - 30