Lab 5: USB audio, Stepper Motor

EE-379 Embedded Systems and Applications
Electrical Engineering Department, University at Buffalo
Last update: Cristinel Ababei, March 2013

1. Objective

The objective of this lab is to learn through examples about:

- USB: play a youtube song through the speaker available on the MCB1700 board and control its volume via the potentiometer also available on the board.
- Control of a simple stepper motor using four GPIOs and the joystick of MCB1700 board to set the direction of the rotation.

2. USB Introduction

USB was developed (in the mid-1990s) by Compaq, Intel, Microsoft, and NEC, joined later by Hewlett-Packard, Lucent and Philips. The USB was developed as a new means to connect a large number of devices to the PC, and eventually to replace the "legacy" ports (serial ports, parallel ports, keyboard and mouse connections, joystick ports, midi ports, etc.). The USB is based on a "tiered star topology" in which there is a single host controller and up to 127 "slave" devices. The host controller is connected to a hub, integrated within the PC, which allows a number of attachment points (referred to as ports). This is illustrated in the figure below.

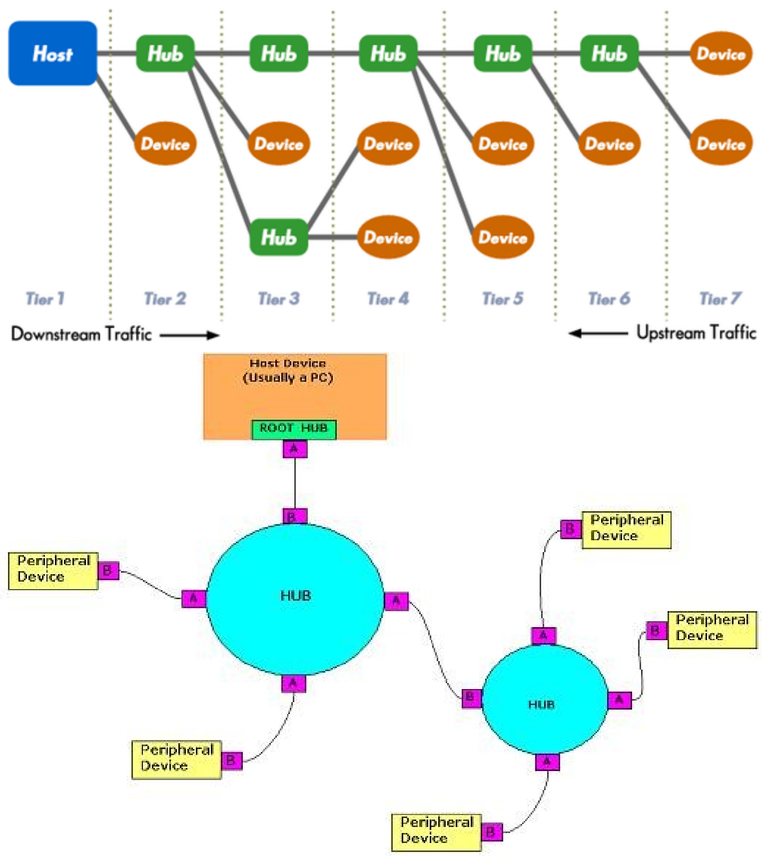

Figure 1 USB "tiered star topology".

USB is intended as a bus for devices near to the PC (up to 5 m). For applications requiring distance from the PC, another form of connection is needed, such as Ethernet.

USB is a four-wire bus (requires a shielded cable containing 4 wires) that supports communication between a host and one or more (up to 127) peripherals. The host controller allocates the USB bandwidth to attached devices through a *token-based protocol*. The bus supports hot plugging and dynamic configuration of the devices. All transactions are initiated by the host controller.
The host schedules transactions in 1 ms frames. Each frame contains a Start-Of-Frame (SOF) marker and transactions that transfer data to or from device endpoints. Each device can have a maximum of 16 logical or 32 physical endpoints. There are four types of transfers defined for the endpoints.

- Control transfers are used to configure the device.
- Interrupt transfers are used for periodic data transfer.
- Bulk transfers are used when the rate of transfer is not critical.
- Isochronous transfers have guaranteed delivery time but no error correction.

LPC1768 has a USB 2.0 interface that can be configured as:

- **Host**
- **Device**
- **OTG**: USB On-The-Go, often abbreviated USB OTG or just OTG, is a specification that allows USB devices such as digital audio players or mobile phones to act as a host, allowing other USB devices like a USB flash drive, mouse, or keyboard to be attached to them.

The USB device controller on the LPC17xx enables full-speed (12 Mb/s) data exchange with a USB host controller. A simplified block diagram of the three USB connectors of the MCB1700 board is shown in the figure below (see also the schematic diagram of the MCB1700 board; file available in archive of lab2).

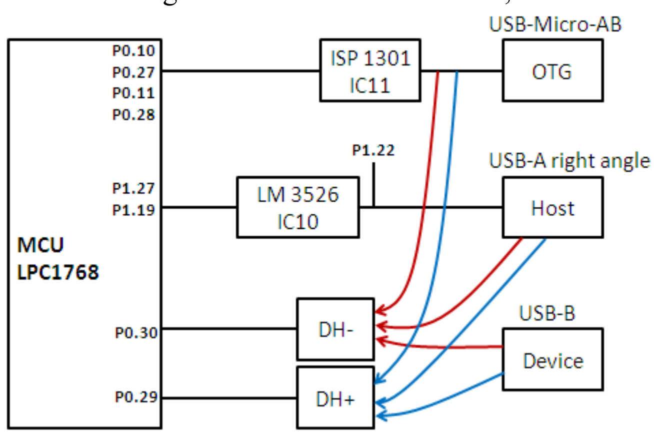

Figure 2 The three USB connectors on MCB1700 board.

Some of the most popular types of USB connectors are shown in figure below:

Figure 3 Common USB connectors.

There is a lot of online information describing the USB. ***As a start, please read the first pointer from the references suggested in [2].*** Also, take a look at Chapters 11,12,13 of the LPC17xx user manual.

3. Example 1: Play audio to speaker of MCB1700

The Audio project example is a demo program for the KeilMCB1700 Board using the NXP LPC17xx Microcontroller. It demonstrates an USB Audio Device – Speaker. The USB Audio Device is recognized by the host PC running Windows which will load a generic Audio driver and add a speaker which can be used for sound playback on the PC. Potentiometer on the board is used for setting the Volume.

The files of this example are in **keil_example/USBAudio/**. Launch uVision and open the project; clean and re-build. Download to the board. Now, start a web browser and play the clip from:
http://www.youtube.com/watch?v=Tj75Arhq5ho
Te audio should play on the speaker of the MCB1700 board. Tune the potentiometer, listen, and enjoy. Observe operation and comment.

This is not an easy project example. However, take some time to read the source code and try to get a birds-eye-view understanding of it. Start with **usbmain.c** and backtrack the function calls. Only with a thorough understanding of the USB theory this source code would be easier to digest.

Optional: Study the USBHID example too.

4. Stepper Motor Introduction

A stepper motor (also referred to as step or stepping motor) is an electromechanical device achieving mechanical movements through conversion of electrical pulses. Stepping motors can be viewed as electric motors without commutators. Typically, all windings in the motor are part of the stator, and the rotor is either a permanent magnet or, in the case of variable reluctance motors, a toothed block of some magnetically soft material. All of the commutation must be handled externally by the motor controller, and typically, the motors and controllers are designed so that the motor may be held in any fixed position as well as being rotated one way or the other.

Stepper motors are driven by digital pulses rather than by a continuous applied voltage. Unlike conventional electric motors which rotate continuously, stepper motors rotate or step in fixed angular increments. A stepper motor is most commonly used for position control. With a stepper motor/driver/controller system design, it is assumed the stepper motor will follow digital instructions. Most steppers can be stepped at audio frequencies, allowing them to spin quite quickly, and with an appropriate controller, they may be started and stopped at controlled orientations.

For some applications, there is a choice between using servomotors and stepping motors. Both types of motors offer similar opportunities for precise positioning, but they differ in a number of ways. Servomotors require analog feedback control systems of some type. Typically, this involves a potentiometer to provide feedback about the rotor position, and some mix of circuitry to drive a current through the motor inversely proportional to the difference between the desired position and the current position.

In making a choice between steppers and servos, a number of issues must be considered; which of these will matter depends on the application. For example, the repeatability of positioning done with a stepping motor depends on the geometry of the motor rotor, while the repeatability of positioning done with a servomotor generally depends on the stability of the potentiometer and other analog components in the feedback circuit.

Stepping motors can be used in simple open-loop control systems; these are generally adequate for systems that operate at low accelerations with static loads, but closed loop control may be essential for high accelerations, particularly if they involve variable loads. If a stepper in an open-loop control system is overtorqued, all knowledge of rotor position is lost and the system must be reinitialized; servomotors are not subject to this problem.

In this lab, we'll use a so called "five wire stepper" shown in the figure below. This 28BYJ48 stepper motor is driven via a driver board that contains 4 Darlington drivers (ULN2003) and 4 LEDs.

Figure 4 Stepper motor with driver board.

The diagram below shows the 5 wires connected to the motor, which must be plugged into the driver board. The connections between the MCB1700 (read the source code of the example in the next section) and the ULN2003 driver board are:

- 5V+ connect to +(5..12)V
- 5V- connect to 0V (Ground)
- IN1: to MCB1700 pin P0.0
- IN2: to MCB1700 pin P0.1
- IN3: to MCB1700 pin P0.2
- IN4: to MCB1700 pin P0.3

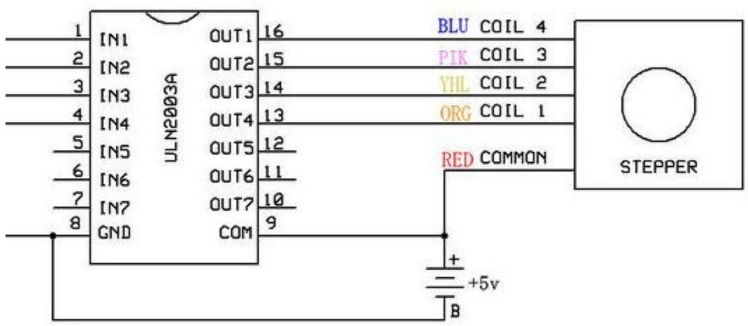

Figure 5 Connections between drive board and stepper.

Figure below shows a closer look at the connections of the stepper motor.

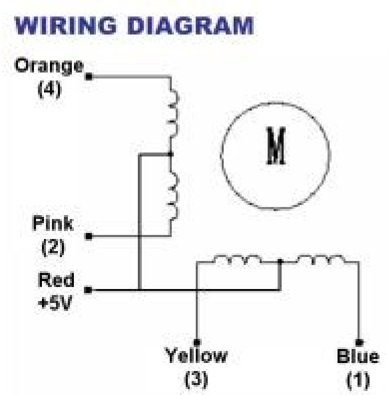

Figure 6 Wiring of the stepper motor.

To drive the motor we set to logic high/low the pins of the MCU in an 8-phase order as shown in the figure below (clockwise movement):

Drive IN4 only
Drive IN4 and IN3
Drive IN3 only
Drive IN3 and IN2
…

Lead Wire Color	---> CW Direction (1-2 Phase)							
	1	2	3	4	5	6	7	8
4 ORG	-	-						-
3 YEL		-	-	-				
2 PIK				-	-	-		
1 BLU						-	-	-

Figure 7 Order of driving the first four wires of the stepper.

For more details on stepper motors take a look at the suggested references [3] and search on the Internet.

5. Example 2: Control of a stepper motor

In this example, we control the rotation of the 28BYJ48 stepper motor via the joystick on the MCB1700 board. If the joystick is pressed to the left the rotation is clockwise and if the joystick is pressed to the right the rotation is anti-clockwise.

The files necessary for this example are located in **lab5_stepper/** folder as part of the downloadable archive for this lab. This is actually the whole uVision project directory. Just copy it to **keil_examples/** (this is the code-bundle directory of lab#2). Then clean and re-build the project. Download to the board. Connect the stepper motor and the driver board as shown in the figure below using the **power supply and the wires provided by your TA**. Observe operation and comment.

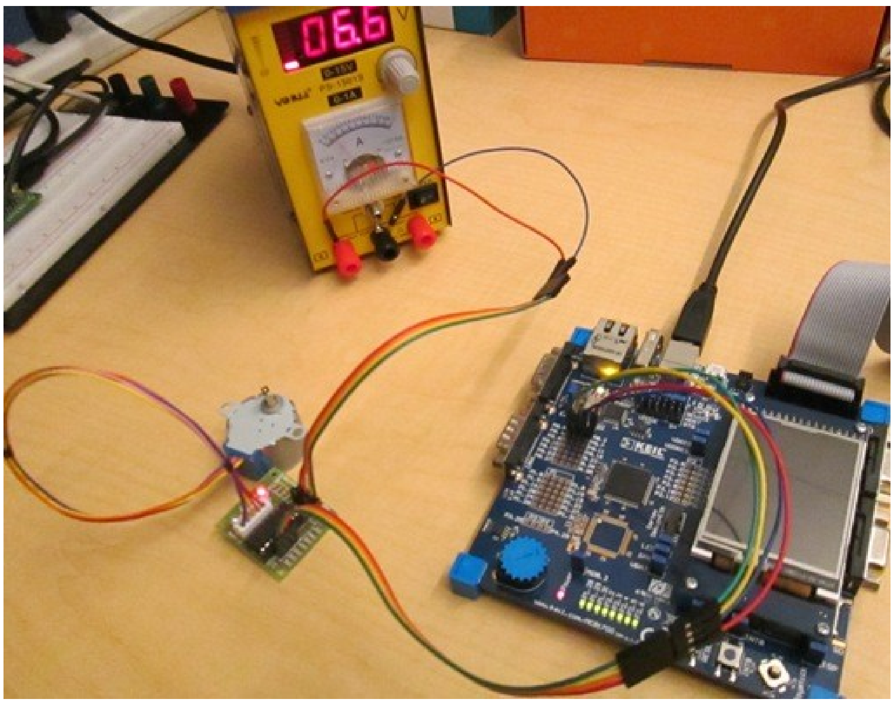

Figure 8 Stepper motor setup.

6. Lab Assignment

This lab has no lab assignment (no pre-lab report, no final lab report). Instead you should use the extra lab-time to work on your projects and on the HW assignment #4.

7. Credits and references

[1] Micro SD(HC) cards:
[2] USB:
--USB Made Simple (excellent read): http://www.usbmadesimple.co.uk/index.html
--USB Background; http://www.totalphase.com/support/kb/10047/
--USB Home: http://www.usb.org/home
-- http://www.keil.com/rl-arm/rl-usbhost.asp

--USB on-the-go (OTG) basics; http://www.maximintegrated.com/app-notes/index.mvp/id/1822
-- http://en.wikipedia.org/wiki/Universal_Serial_Bus
--LUFA; http://www.fourwalledcubicle.com/LUFA.php
[3] Stepper motors:
--Control of stepping motors, a tutorial; http://homepage.cs.uiowa.edu/~jones/step/
--Sparkfun tutorial; https://www.sparkfun.com/tutorials/400
--More tutorials; http://www.stepperworld.com/pgTutorials.htm
--Even more tutorials; http://www.epanorama.net/links/motorcontrol.html#stepper
--Stepper Motor 5V 4-Phase 5-Wire & ULN2003 Driver Board (includes detailed specs); http://www.geeetech.com/wiki/index.php/Stepper_Motor_5V_4-Phase_5-Wire_%26_ULN2003_Driver_Board_for_Arduino
Buy it for $2.02 (amazing price) on Amazon: http://www.amazon.com/28BYJ-48-28BYJ48-4-Phase-Arduino-Stepper/dp/B0089JV2OM/ref=pd_sim_sbs_vg_1