Warsaw 18.05.2018

Linux for Embedded Systems – Laboratory ex. 4

The student(s): Patryk Walczak

Description of the assignment

The subject of the assignment 4 is the implementation with the emulated Vexpress A9 board a device equipped with two forms of user interface:

1. simple buttons and LEDs should be used to control basic functions;

2. web interface (or other network interface) should be used to control advanced functions;

3. The design should include support for additional equipment connected to the board. In the simplest version it can be a built-in audio output. In a better version it can be an external USB audio card or other device. Please remember that QEMU allows you to share USB devices with the emulated board.

I chosed the suggetion:
D) Using the standard audio output of the RPi, please build the music player. The local buttons and LEDs should be used to switch on/off the player, to select the song and to control volume. The web interface should be used to cofigure the list songs, to download and to remove the song.

Procedure to recreate the design from the attached archive

I prepared:
- build.sh – script which downloads and prepares the configuration
- run_before – script which runs SwitchDemo
- runme – script which runs QEMU and starts buildroot

To reproduce my script:
1. run build.sh script
2. run run_before script
3. run runme script
4. open browser and write address http://localhost:2222/ after the runme script ends its task

Description of the solution

For the solution, I based on gpio_simpe_USB and I had to add:
Target packages → Audio and video applications → alsa-utils → alsamixer and speaker-test
Target packages → Audio and video applications → mpd and mpd-mpc
Target packages → Audio and video applications → mpd → curl
Target packages → Audio and video applications → mpd → libsoxr and libsamplerate
Target packages → Audio and video applications → mpd → all decoder plugins except tremor
Target packages → Interpreter languages and scripting → python3 and:
→ internal modules : readline, ssl, sqlite, zlib, xml, xz
→ external modules : python-flask, python-flask-babel, python-flask-login, python-wtforms, websockets
Target packages → Networking applications → ifplugd
Target packages → Shell and utilities → screen

Those packages are required in my solution.
Doing the task I based on my previous solutions.

Buttons and LEDs controlled I had to transform from C++ code to the Python code. But the transformation is not perfect and the current version is no perfect. One

fast click on a button is to short for my algorithms which should handle the „bounce effect”. Each time you would like to use the button please hold the button for one second. To start/stop music use button 12, to play the next song hold button 13. To turn up the volume click button 14 and 15 to turn down.

In the case of the web service, I changed the upload and download endpoint for working with mp3 files. Additionally, I add „/up” endpoint to move a song up in the playlist and „/down” which move song one position down in the playlist. The endpoints are achievable but I made a form in the main path „/”. In the form user has to write the full name of the song and press „Up” or „Down” button to move the song in queue. The logging utility still works but I require the nickname and password only for uploading and downloading song.

The main difference in the solution is using of the Music Player Daemon. With the utility, the buildroot plays the music and with the python functions the control is made in my applications. Each application (web service and music_player) has own MPDClient. All actions are performed using the clients. But the connection to the daemon sometimes is lost, and for this purposes, the „reconnection()” function is in both programs. I think it is a better solution than disconnecting each time. If the connection is active the function only checks it otherwise connect again to the MPD.

The port 80 from qemu is forwarded to 2222 port in localhost. So for establishing a connection with the web service, the address is http://localhost:2222/.