Warszawa 26.04.2024

Linux w Systemach Wbudowanych – Laboratorium ćw. 2

Student: Jakub Kępka

Treść zadania:
1. Zaimplementowanie aplikacji w języku kompilowanym (najlepiej C) obsługującej przyciski i diody LED. Aplikacja powinna reagować na zmiany stanu przycisków bez oczekiwania aktywnego. Na zajęcia wykonałem aplikację, która sumuje dwie liczby binarne. Sekwencja aplikacji jest następująca: aplikacja czeka na liczbę nr 1, która wprowadzana jest za pomocą sw1 i sw2 (sw1 to 0, sw2 to 1) -> sw3 potwierdza wpisanie liczby 1 -> aplikacja czeka na drugą liczbę, która jest wprowadzana analogicznie -> sw3 potwierdza wpisanie drugiej liczby -> liczba wynikowa jest wyświetlana za pomocą diod, analogicznych do przycisków.

2. Przetestowanie korzystania z debugger'a (gdb) przy uruchamianiu aplikacji.

3. Przekształcenie aplikacji w pakiet Buildroot'a.

Specyfikacja aplikacji:

Aplikacja jest grą w papier, kamień i nożyce.
Jednym graczem jest użytkownik, drugim jest komputer. Użytkownik klika jeden z przycisków, każdy przycisk reprezentuje papier, kamień lub nożyce.
W momencie, gdy użytkownik kliknie dany przycisk komputer losuje przedmiot którym odpowiada. W zależności od wyniku rundy zapala się odpowiednia dioda (wygrana, przegrana lub remis). Aplikacja włączona jest w nieskończonej pętli, po rozegranej rundzie następuje kolejna. Między czasie zliczane są wyniki każdej rundy.

Opis rozwiązania:
Aplikacja została wykonana przy pomocy biblioteki c-periphery. W zdaniu korzystałem z dokumentacji dostarczoną na stronie https://github.com/vsergeev/c-periphery/tree/master. Aby utworzyć pakiet buildroota utworzyłem dwa foldery, jeden poziom wyżej od buildroota z plikami SRP.c oraz Makefile, drugi w folderze package w buildroot o nazwie SRP z plikami Config.in oraz SRP.mk.

Przekształcanie aplikacji w paczkę:
Tworzymy folder naszego pakietu dla danej aplikacji. Tworzymy go w folderze packages. Dodajemy do niego plik config.in oraz plik makefile. Zarys treści tych plików wziąłem z wykładu i przerobiłem według swoich potrzeb. Między innymi dodałem potrzebne dependencies.

Debugger:
Kompilacja odpowiednich pakietów:
Target packages → Debugging → gdb → gdbserver
Toolchain → Build cross gdb for the host

Włączyłem aplikację na systemie docelowym

gdbserver :7654 /SPR

Opis modyfikacji i konfiguracji Buildrota i kernela:

Konfiguracja jak na poprzednim laboratorium, dodatkowo:
Wstępna konfiguracja pochodzi z pliku raspberrypi4_64_defconfig, następnie zostały ustawiona konfiguracja z poradnika. Ponadto ustawione zostały opcje: Target packages/Libraries/Hardware handling -> c-periphery
Target packages → Debugging → gdb → gdbserver
Toolchain → Build cross gdb for the host