



Katedra Systemów Geoinformatycznych

# Aplikacje Systemów Wbudowanych

Laboratorium

część 3

ćwiczenia 10-11



Politechnika Gdańska

Wydział Elektroniki, Telekomunikacji i Informatyki  
Katedra Systemów Geoinformatycznych

Gdańsk 2010

## Wprowadzenie

W tej części laboratorium studenci zapoznają się ze środowiskiem programistycznym Qt oraz z zestawem JUPITER 5.7 VGA opartym o procesor Freescale i.MX31. Środowisko Qt jest bardzo przenośne – wspiera takie platformy jak:

- Windows
- Linux
- Windows CE
- Embedded Linux
- Symbian

Przed rozpoczęciem pracy proszę zapoznać się z poniższym opisem stanowisk laboratoryjnych. Ćwiczenia realizowane są w grupach 2-osobowych.

## Konfiguracja laboratorium

Na potrzeby ćwiczenia z platformą Jupiter została przygotowana płyta zawierająca zmodyfikowany system operacyjny Ubuntu 9.10. System uzupełniono o:

- Pakiet Qt Embedded skompilowany pod architekturę ARM,
- Dokumentację Qt w formacie HTML,
- Kompilator C++ Code Sourcery dla architektury ARM,
- System plików dla Embedded Linux (/JupiterFS),
- Serwer NFS udostępniający system plików dla EL przez sieć.

System z płyty konfiguruje interfejs sieciowy na adres 192.168.1.1, natomiast urządzenia skonfigurowane są w zakresie (192.168.1.10-20). Do uruchomienia systemu Linux na platformie Jupiter konieczne jest połączenie sieciowe (kabel skrosowany) z komputerem PC ponieważ na potrzeby laboratorium Jupitera zostały skonfigurowane do uruchamiania systemu udostępnionego przez serwer NFS. Uruchamiane aplikacje obsługiwane będą przy użyciu ekranu dotykowego – emulacji myszy.

## Kompilacja aplikacji w środowisku Qt dla ARM

Podstawowe informacje na temat projekt utworzonego w środowisku QtCreator znajdują się w pliku <nazwa>.pro. Aby skompilować na jego podstawie cały projekt należy:

1. uruchomić polecenie `qmake`, podając jako parametr docelową architekturę; polecenie `qmake` stworzy standardowy plik `Makefile`;

```
qmake -o Makefile <nazwa>.pro
```

zazwyczaj linię tą należy uzupełnić o specyfikację platform docelowej, np:

```
-spec /usr/local/Trolltech/QtEmbedded-4.5.3-arm/mkspecs/qws/linux-arm-g++  
Ale w przygotowanym systemie ustawiono zmienną środowiska QMAKESPEC.
```

2. uruchomić polecenie **make**, które zbuduje aplikację.

W systemie przygotowanym na płycie CD ścieżki dostępu do Qt i kompilatora są odpowiednio ustawione.

## Uruchamianie aplikacji

Jądro systemu Embedded Linux jest umieszczone w pamięci Flash urządzenia, natomiast system plików jest podłączany z komputera PC za pośrednictwem NFS. Aby umożliwić uruchomienie własnej aplikacji – należy umieścić ją w udostępnianym systemie plików: `/JupiterFS/laboratorium`

Następnie w pliku `/JupiterFS/laboratorium/autostart.sh` należy dopisać polecenie uruchomienia aplikacji. Możesz uruchomić kilka aplikacji pod rząd – po zamknięciu jednej uruchomiona zostanie kolejna.

## Ważne pliki i katalogi

System plików PC	System plików Jupiter	Opis
<code>/JupiterFS</code>	<code>/</code>	Główny katalog systemu plików udostępnianego z PC platformie Jupiter
<code>/JupiterFS/laboratorium</code>	<code>/laboratorium</code>	Katalog roboczy
<code>/JupiterFS/laboratorium/autorun.sh</code>	<code>/laboratorium/autorun.sh</code>	Plik uruchamiany zaraz po starcie systemu platformy Jupiter.
<code>/usr/local/Trolltech/</code>	---	Katalog z Qt skonfigurowanym pod architekturę ARM
<code>/usr/CodeSourcery</code>	---	Kompilator C++ do ARM
<code>/laboratorium/</code>	---	Katalog z materiałami do laboratorium
<code>/laboratorium/QtDoc</code>		Dokumentacja Qt

## Ćwiczenie 10.

### Zadanie 1. Uruchomienie platformy Jupiter

Uruchom system operacyjny z płyty CD, zaczekaj na pojawienie się shella. Następnie uruchom platformę Jupiter. Po prawidłowym wykonaniu na ekranie platformy Jupiter powinien uruchomić się zegar.

Przeanalizuj zawartość katalogu `/JupiterFS/laboratorium/` i pliku `autorun.sh`

Źródła programu znajdują się pod linkiem `/laboratorium/clock`

W przypadku kłopotów z uruchomieniem przykładu – poproś prowadzącego o pomoc.

### Zadanie 2. Uruchomienie i modyfikacja przykładów

Uruchom kilka z przykładów pakietu Qt. Przykłady znajdują się w katalogu `/laboratorium/examples`. Aby uruchomić gotowy program wystarczy, że dopiszesz jego wywołanie do pliku `autorun.sh` i uruchomisz ponownie urządzenie.

Następnie skompiluj wybrany przykład, modyfikując go tak, aby zajmował pełen ekran Jupitera. Kolejne kroki to:

1. Stworzenie pliku Makefile dla architektury arm – polecenie `qmake` z odpowiednim parametrem,
2. Skompilowanie programu – `make`,
3. Przeniesienie binariów do katalogu udostępnianego za pośrednictwem NFS,
4. Dopisanie programu do skryptu uruchamianego przy starcie platformy,
5. Uruchomienie platformy Jupiter.

Szczegółowe informacje na temat każdego z kroków znajdziesz we wprowadzeniu do laboratorium.

### Zadanie 3. Kompilacja szablonu

Skompiluj program znajdujący się w katalogu `/laboratorium/malarz`

Uruchom go na platformie Jupiter. Następnie zmodyfikuj kod (odkomentuj odpowiedni fragment), tak aby na jednym formularzu umieszczone były dwa widgety Qt. Uruchom zmodyfikowany program.

**Ważne!** W swojej podstawowej wersji program `malarz` zawiera jedno główne okno, z jednym widgetem (w innych językach np. kontrolka), który reaguje na kliknięcie myszą (reimplementacja chronionej metody `mousePressEvent`) i potrafi się sam odrysować (`paintEvent`). Po modyfikacji mamy dwie niezależne instancje tego samego widgetu.

### Zadanie 4. Dodatkowe funkcje rysowania

Zmodyfikuj szablon uzupełniając o dodatkową funkcjonalność:

- Druga figura  
po naciśnięciu drugiego guzika myszy – rysuj w tym miejscu prostokąt, zachowanie identyczne jak koła (ale koło ma nadal działać na lewy klawisz);
- Wypisywanie aktualnej pozycji kursora  
Zapamiętuj pozycję kursora w stworzonym przez siebie atrybucie, uaktualniaj ją w `mouseMoveEvent()` i wypisuj w `paintEvent`;

- Rysowanie kół o dowolnym rozmiarze

Na naciśnięcie guzika powinien zostać uruchomiony tryb rysowania, ale dopóki palec dotyka ekranu – można powiększać koło (nie trzeba obsługiwać zmniejszania obszaru!). W metodzie `MouseEvent` uruchom tryb rysowania, modyfikuj promień i wołaj `Update` w `MouseMoveEvent` i kończ rysowanie w `MouseEvent`.

- Zmiana koloru

Stwórz nowy widget (nowa klasa) **CKolory** – opierając się na `CPlotno`. Zmodyfikuj: nazwę i rozmiar (na np. 50 x 320). W metodzie `paintEvent` narysuj 6 prostokątów o różnych kolorach. W metodzie `mousePressEvent` musisz na podstawie współrzędnej Y określić kolor, a następnie umożliwić użycie go w kodzie klasy `CPlotno`.

Proponowane rozwiązanie: założenie, że będzie tylko jedna instancja `CKolory` i użycie statycznego atrybutu klasy. W definicji (.h):

```
public:
    static QColor kolor;
```

A w pliku `cpp` deklaracja:

```
CKolory::kolor = Qt::white;
```

Przy samym rysowaniu w `Cplotno` po prostu skorzystaj z `CKolory::kolor` (pamiętając o `include`).

## Ćwiczenie 2.

Uruchom przygotowaną aplikację, po zatwierdzeniu tematu przez prowadzącego na pierwszych zajęciach. Przykładowe tematy to:

- klawiatura sms,
- Paint: 16 kolorów, rysowanie linii, okręgów, prostokątów,
- Podgląd obrazka (bez zoom – z przesuwaniem),
- Podgląd obrazka (zoom – bez przesuwania),
-

## Dodatek A

### Opis układu ADELAIDE

Układ ADELAIDE jest niewielkich rozmiarów modułem typu SBC (Single Board Computer) wyposażonym w 32-bitowy procesor Freescale i.MX31 bazujący na jądrze ARM11.

Wykorzystanie wspomnianego mikroprocesora skutkuje uzyskaniem korzystnego stosunku mocy obliczeniowej do zużywanej energii. Dodatkowo uzyskana moc obliczeniowa również dzięki zastosowaniu dedykowanych, zintegrowanych podukładów jest wystarczająca do efektywnego przeprowadzania wymagających operacji związanych z dekompresją/kompresją video.

Podstawowymi elementami i cechami platformy ADELAIDE są:

3. Mikroprocesor 532 MHz Freescale i.MX31 oparty na architekturze ARM11
4. Zintegrowany układ SVGA-GPU (800 x 600)
5. Pamięć RAM
6. Pamięć Flash (Nand)
7. Możliwości multimedialne:  
dekompresja/dekodowanie MPEG-4  
AC97 audio codec
8. Niskie zużycie prądu 600 mA
9. Szeroki zakres temperatur -30° ~ +85° C
10. Wsparcie dla systemów czasu rzeczywistego Windows CE oraz Linux.

### Opis platformy JUPITER 5.7 VGA

Platforma JUPITER 5.7 VGA jest rozbudowanym o układy wejścia/wyjścia i interfejsy użytkownika rozwiązaniem dedykowanym dla układu ADELAIDE.

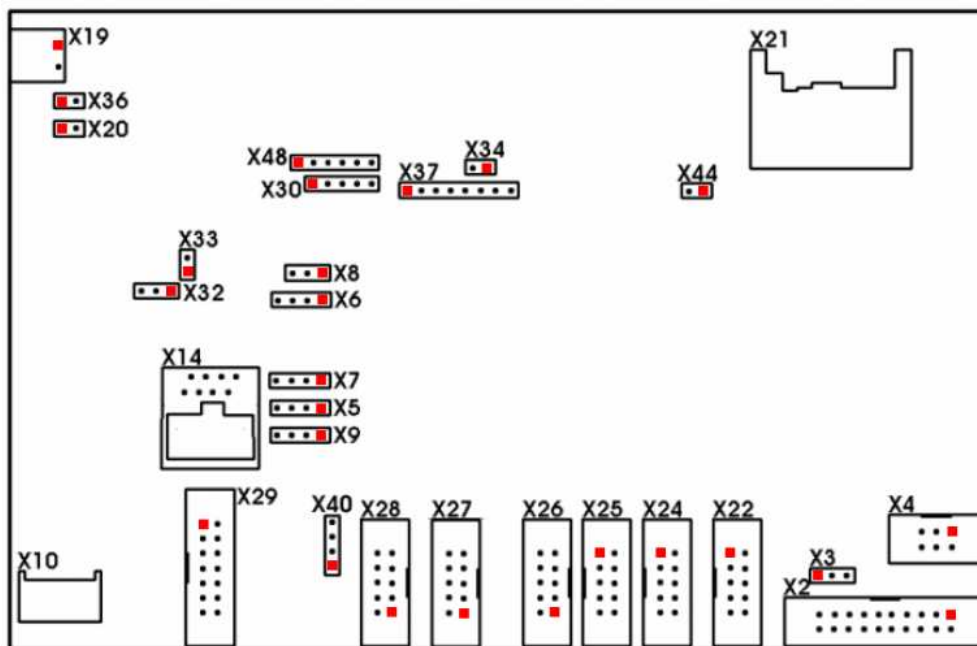
Dzięki Platformie JUPITER 5.7 VGA uzyskujemy dostęp do następujących interfejsów procesora Freescale i.MX31:

- RS-232,
- USB,
- CAN,
- SD/MMC,
- SDIO,
- A/D I/O,
- I<sup>2</sup>C,
- SPI,

Ponadto z wykorzystaniem podukładów wspomagających zapewnia dostęp do interfejsów:

- Ethernet,
- Audio In/Out.

Na rys. 1 przedstawione zostały złącza wyprowadzenia poszczególnych interfejsów.



X 2	Analog input	X 24	RS232 – 2
X 3	Analogue reference voltage selector	X 25	RS232 – 3
X 4	Analog output	X 26	SPI – 1
X 5	Audio Line out	X 27	SPI – 2
X 6	Audio Line in	X 28	SPI – 3
X 7	Audio Headphone out	X 29	Digital I/O
X 8	Audio Microphone in (mono)	X 30	USB HighSpeed Host
X 9	Audio S/PDIF out	X 32	CAN connector
X 10	Backlight inverter output	X 33	CAN terminator
X 11	n/a	X 34	JTAG enable signal for CPU
X 14	Ethernet RJ45	X 36	Reset switch
X 19	Power supply, Phoenix connector	X 37	JTAG interface
X 20	Powerswitch	X 40	I <sup>2</sup> C interface
X 21	MMC/SD-Card socket	X 44	Clear all
X 22	RS232 – 1	X 48	USB HighSpeed OTG

Rys. 1. Schemat i opis wyprowadzeń na platformie Jupiter 5.7 VGA

## Możliwości programowania systemu JUPITER 5.7 VGA + ADELAIDE

Powyższa platforma wspiera systemy operacyjne przeznaczone do systemów wbudowanych:

- Windows CE,
- Embedded Linux.

Dla systemu Windows CE dostępne są sterowniki większości urządzeń znajdujących się na platformie między innymi:

- UART
- Ethernet (sterownik 10/10 Mbit, dla układu SMCS 9215)
- USB Host (wsparcie dla urządzeń gromadzenia danych (USB-stick, hard disc, CD-ROM..., również klawiatura, myszka, USB hub itp.)
- USB Client (dla połączeń typu ActiveSync)
- Audio 16Bit wyjście stereo (do to 48kHz), mic-in, line-in

- Touch Screen (wsparcie dla typu urządzenia dotykowego z interfejsem typu: 4-wire resistive screen)
- Wyświetlacz (wsparcie dla wyświetlacza Kyocera TCG057VG1AC zawartego w zestawie startowym)
- Sterownik SD/MMC Karty pamięci (MMC, SD, SDHC) oraz urządzenia SDIO jak Wireless LAN, Bluetooth, GSM, GPS
- Flash File System Standard FAT, TFAT (transaction safe FAT).

Wsparcie dla systemu Linux jest bardziej ograniczone i obejmuje następujące elementy:

- UART
- Ethernet (sterownik 10/10 Mbit, dla układu SMCS 9215)
- USB Host (wsparcie dla urządzeń gromadzenia danych (USB-stick, hard disc, CD-ROM..., również klawiatura, myszka, USB hub itp.) **(Ograniczone możliwości oraz liczne błędy)**)
- Wyświetlacz (wsparcie dla wyświetlacza Kyocera TCG057VG1AC zawartego w zestawie startowym)
- Touch Screen (wsparcie dla typu urządzenia dotykowego z interfejsem typu: 4-wire resistive screen) – do zestawu dołączone urządzenie zintegrowane z wyświetlaczem Kyocera TCG057VG1AC
- Sterownik SD/MMC Karty pamięci (MMC, SD, SDHC)
- Flash File System: **JFFS2 (Journalling Flash File System version 2)**.

Obecnie, dla platformy testowej, dostępne są wersje jądra systemu operacyjnego Linux generacji:

- 2.6.23 (najdłużej rozwijana, dostępna rewizja 2.6.23.95),
- 2.6.25 (nie wykorzystywana w projekcie iCOM, dostępna rewizja 2.6.25.10),
- 2.6.26 (najnowsza gałąź).

### Środowisko deweloperskie

Środowisko rozwojowe aplikacji w systemie Linux dla platformy JUPITER 5.7 VGA + ADELAIDE obejmowało komputer PC z zainstalowanym następującym oprogramowaniem:

- System operacyjny Linux,
- Serwer TFTP,
- Krzyżowy kompilator dla architektury ARM: Sourcery G++ Lite 2009q3-67 for ARM GNU/Linux
- Serwer NFS
- Biblioteka QT oraz QT Creator dla tworzenia aplikacji w przenośnym środowisku QT
- Biblioteka QT Embedded