Main Challenges on implementing Software Defined Radio on General Purpose Processors

Alexandre de Baynast

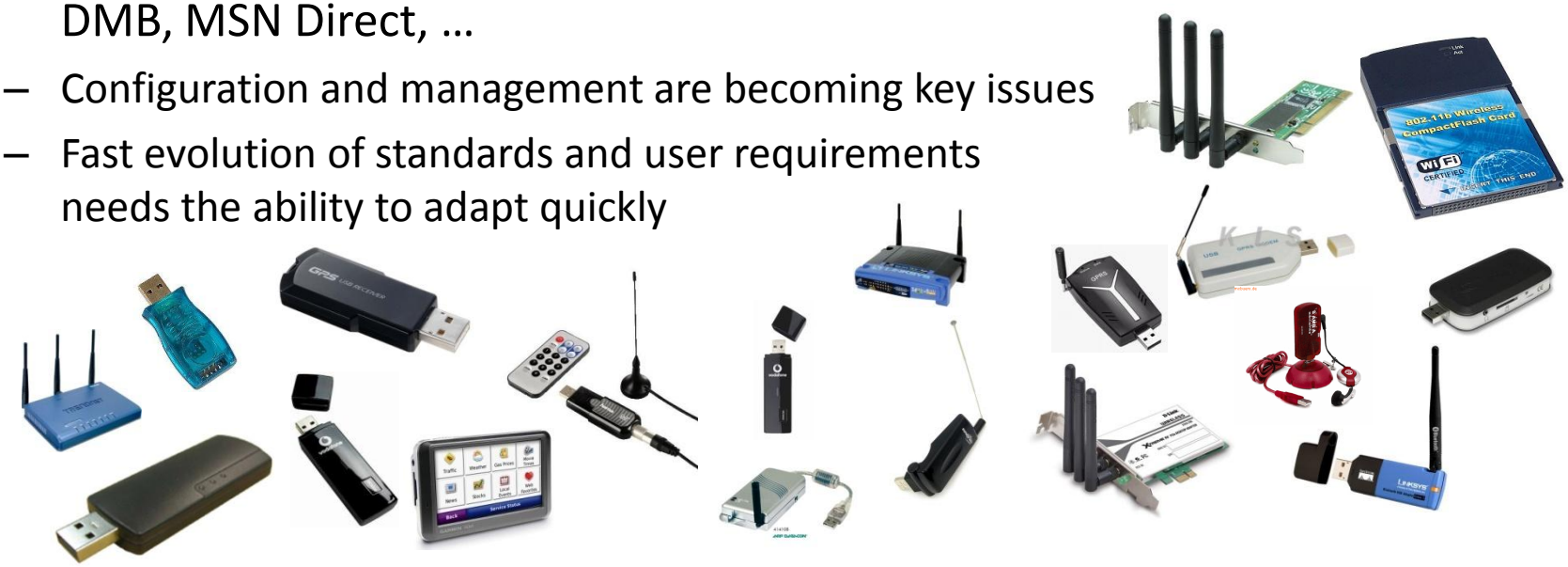
Aachen, March 13, 2009

Freunde Förderer ComNets

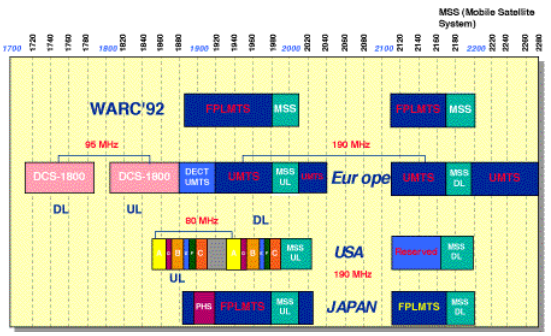
Microsoft Innovation Center Europe

Motivations

- **Rapid evolution of radio standard**
  - GSM, UMTS, 802.11[abgn…], WiMAX, Bluetooth, UWB, GPS, DVB-[TH], DMB, MSN Direct, …
  - Configuration and management are becoming key issues
  - Fast evolution of standards and user requirements needs the ability to adapt quickly



- **Specific regional deployments**
  - Spectrum is allocated by country/continent
    - (FCC in the USA…)



Source: International Telecommunications Union (ITU)

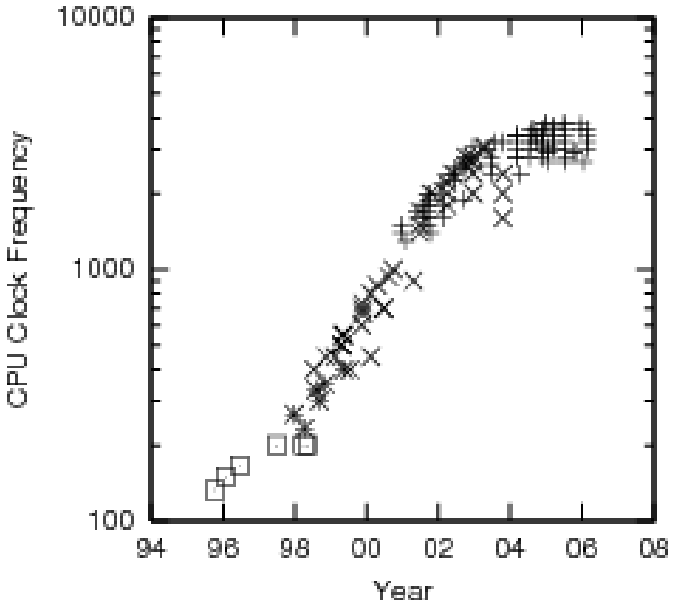
Motivations (cont'd)

- **1st generation: Multiple Hardware devices**
  - Large and fast growing number of radios
  - Device manufacturers often under pressure
    - Heavily dependent on hardware vendors
  - Bad end customer experience
    - They have to purchase new hardware to upgrade to new standards
- **2nd generation: FPGA or multiple DSPs –based Hardware**
  - Expensive Hardware (sample production)
  - Require development of specific software tools like synthesizers
    - SysGen (Xilinx), LabView FPGA Module (NI), Small Form Factor SDR (TI)

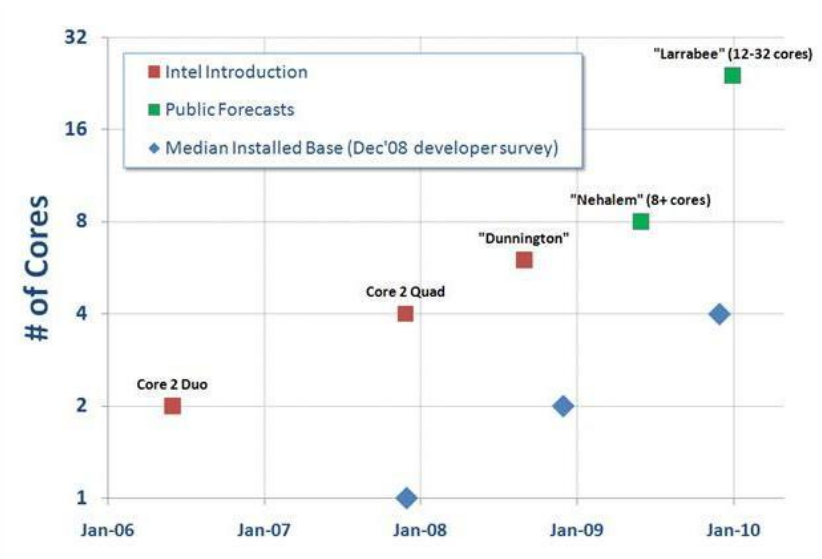
- **3rd generation: Generic Front-End + Software running on computer**
  - Real business model nowadays (Vanu Corp., …)

New context

- **After years of Moore's law, processor industry is now moving multi-core architectures**
  - Sufficient computational power to implement radio standards in software



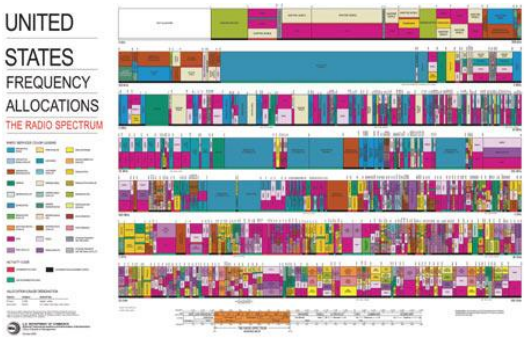
Source: Linux Journal, Jan. 2007



Source: Public forecasts from Intel

New context (cont'd)

- **Under-utilization of radio spectrum**
  - Regulators are now discussing how to assign and use existing spectrum more efficiently (ISM, Whitespaces)



Source: Discover Magazine, June 2007

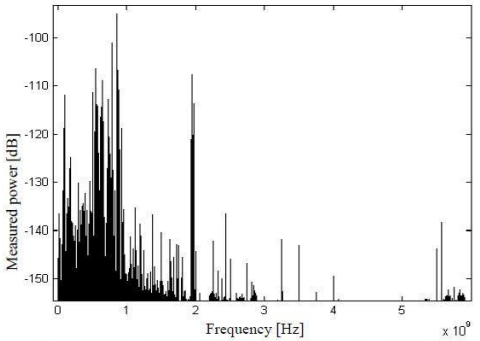
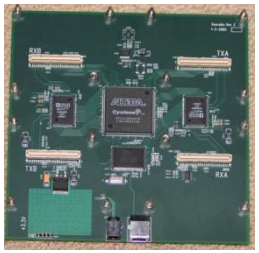


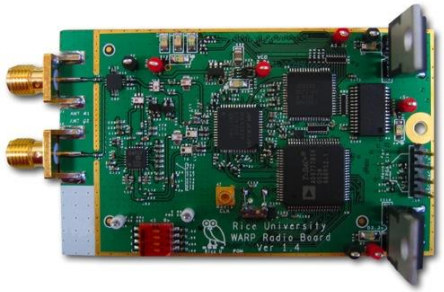
Figure 1. Spectrum utilization measurement at BWRC

Source: Berkeley Wireless Research Center

- **New developments in design of RF Front-end advanced hardware**
  - Supporting large tuning range
  - High-speed bus



Source: Ettus Corporation



Source: Rice University

Research on SDR @EMIC

Introduce new solutions to make future versions of Windows easy-to-use and powerful platforms to manage and use radios:

1. Providing generic building blocks for fast development for manufacturers
2. Seamless optimization of radio resources according to user needs (application level)
3. Decisions on optimizing radio resources need to be taken by GPP on OS level
4. (Windows) SDR has a bright future if we can efficiently exploit the multi-core machines

A simple recipe of Windows SDR implementation for everybody

- **Ingredients:**
  1. Flexible radio front-end which includes a driver to transfer data between GPP and the board
  2. A cable or a bus + a good driver to transfer the data from GPP to the radio
  3. A bunch of basic C++ libraries (filter, demodulator, down-conversion, PLL, Viterbi decoder, CRC, OFDM generator, FFT, convolution, differentiator, integrator, trigonometric operations, over/down-sampling, etc.)


+

+


Proof of concept: RDS-software receiver

- **Radio Data System** (**RDS)** is a communication protocol for sending some digital information in FM signals
  - Type of information: Country code, Radio name, Program type, Alternative Frequencies

- **Coming soon:** Alternative frequency algorithm

USRP Board (HW)

- USRP is a high-speed USB 2.0 based board provided by ETTUS Research LLC for making software radios.

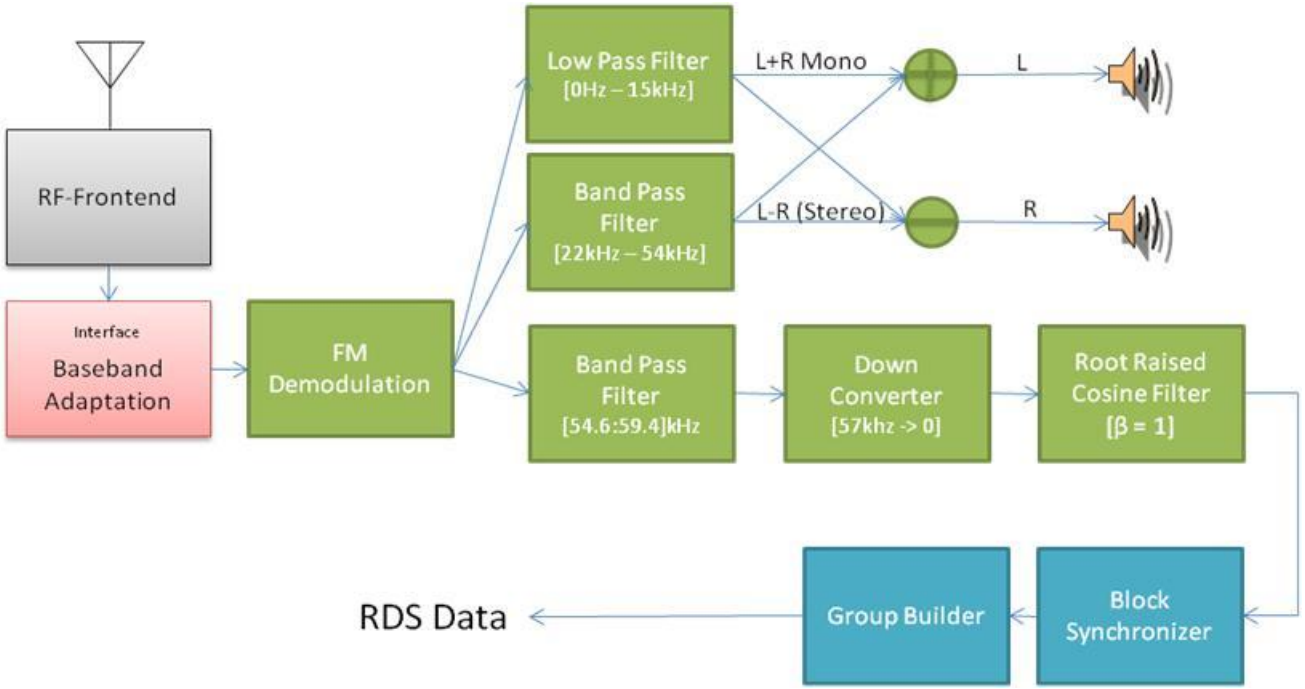
- It consists of:
  - Four high-speed A/D converters (for down-conversion)
  - Four high-speed D/A converters
  - FPGA

- Various plug-on daughterboards.
  - Used BasicRX board for receiving FM transmission
  - BasicRX board supports frequencies from 1MHz to 250MHz

USRPRX

- Software component provided by Trinity College in IRIS framework.
- Reads In-Phase/Quadrature (I/Q) streams from USRP hardware through USB 2.0 interface.
- Down-conversion to baseband is done in FPGA in USRP hardware.
- Parameters:
  - Sampling rate for the baseband signal: 250 KS/s (decimation factor: 256)
  - One USRP block contains 16K Samples

Software implementation of RDS receiver



Performance evaluation

- 2 month development (4PM)


- About 20 basic components (C++ classes)

- MSN Directband runs in real-time on Talladega Platform and Ebox VIA C7 (single-core, 500 MHz, Windows XP)

  CPU Usage: 77% (Talladega), 60% (Ebox)

Live Demo: RDS-software receiver

- **Radio Data System** (**RDS)** is a communication protocol for sending some digital informations in FM signals
  - Similar to DirectBand (Reuse of 15 over 20 components from DirectBand)
  - Type of information: Country code, Radio name, Program type, Alternative Frequencies

- **Coming soon:** Alternative frequency algorithm

Next steps

- **Implementation of broadcast standards with higher data rate**
  - DAB: 1.536 MHz (1.5Msymbols/sec x 4 x 2 x 4B ≈ 400Mb/sec)
  - DVB-T: 8 MHz

- **Implementation of two-way communication standards**
  - Latency issues with current boards

- **Parallelize Processing**
  - Using Data Parallelism to Program GPUs for General-Purpose Uses (Accelerator project at MSR)

Dissemination

- Demo presented during the IEEE SPS symposium on Cognitive Radio

- We are looking for students

(HiWi, Master Thesis, Internship)

http://www.microsoft.com/emic/joboffers.mspx