Getting the first Open Source GSM stack in Linux

Marcin Mielczarczyk <marcin.mielczarczyk@tieto.com>
Krzysztof Antonowicz <krzysztof.antonowicz@tieto.com>

Tieto

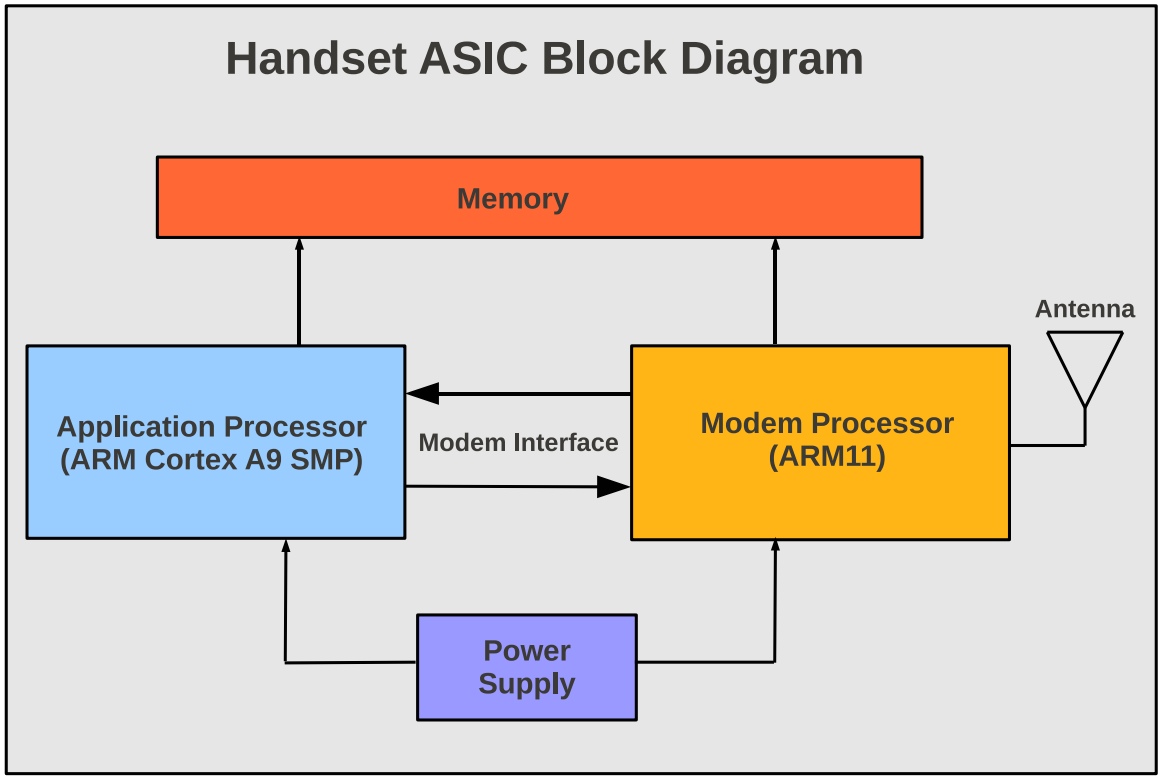
Embedded Linux Conference 2012, Redwood Shores, CA

Outline

1. Introduction
   - Open Source GSM stack
   - Mediatek platform
   - Sciphone Dream G2
2. Porting Linux to new platform
   - HW reverse engineering
   - Executing own code
   - Porting U-Boot
   - Porting Linux
3. Running GSM RF parts
   - GSM RF schematics
   - Description of GSM RF parts
   - DSP reverse engineering
   - Yet to do
4. Summary

Introduction

- Don’t we already have open source mobile phones?



Open Source GSM stack

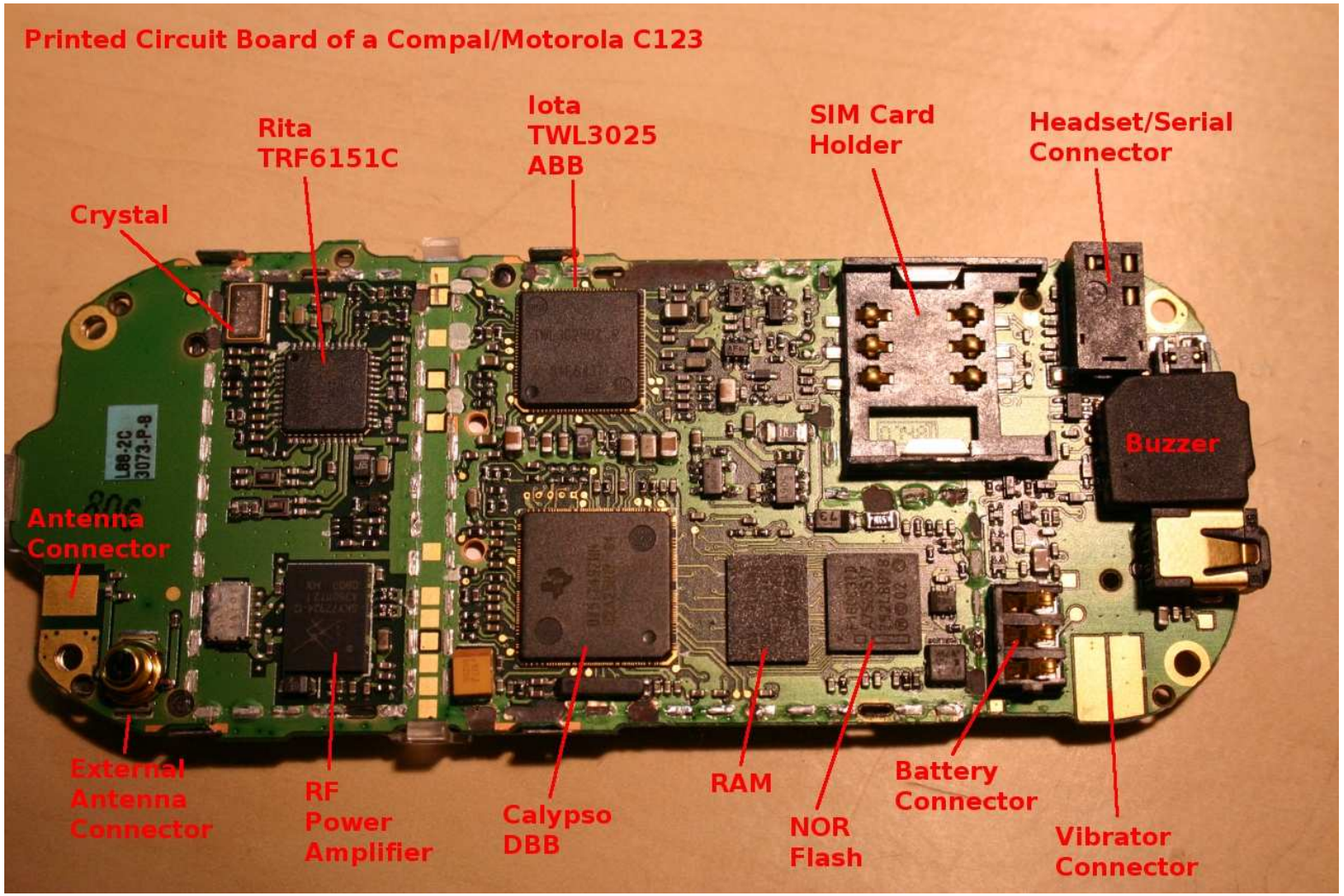
Are there any open source GSM stacks?

OsmocomBB

OsmocomBB (Open Source MObile COMmunication Base Band)

- Supported phones: Compal/Motorola C11x, C12x, C13x, C14x and C15x
- Calypso DBB based on ARM7TDMI
- GSM L1 runs on a mobile phone
- GSM L2/L3 runs on a PC
- It's possible to make GSM Voice calls
- No new phones with Calypso since about 2008

OsmocomBB


Printed Circuit Board of a Compal/Motorola C123
Rita
TRF6151C
Iota
TWL3025
ABB
SIM Card
Holder
Headset/Serial
Connector
Crystal
Buzzer
Antenna
Connector
External
Antenna
Connector
RF
Power
Amplifier
Calypso
DBB
RAM
NOR
Flash
Battery
Connector
Vibrator
Connector

Mediatek platform

Low cost phones based on Mediatek platform

- Feature phones which are mainly fakes of known brands (Nokla, Sany-Ericssan, sciPHONE)
- Most popular SoC is MT622x (based on ARM7TDMI)
- A lot of peripherals included (camera, analog TV, FM transmitter)
- Available and cheap ($30 - $100)
- Platform acquired from Analog Devices

Mediatek platform

ARM7TDMI is not our target, we want to run Linux to have more possibilities. Does Mediatek have something better?

Sciphone Dream G2

Sciphone Dream G2 running fake Android

- Fake of HTC Dream G2 (released before HTC)
- Running Nucleus RTOS with UI like Android
- Based on MT6235 SoC (ARM926EJS)
- Resistive touch screen, WiFi, BT, FM radio, USB, SD/MMC
- Low cost, starting from $50
- Other devices available with MT6235 SoC

MT6235

MT6235 characteristics:

- Single core ARM926EJ-S 208MHz
- Advanced DSP functionality
- PMU / Touch panel driver intergated
- SD/MMC and SDIO support
- Built in USB2.0
- USIM support
- EDGE class 12, GPRS class 12
- Highly integrated (DBB and ABB in one chip)
- Datasheet easily available on the Internet

How to begin

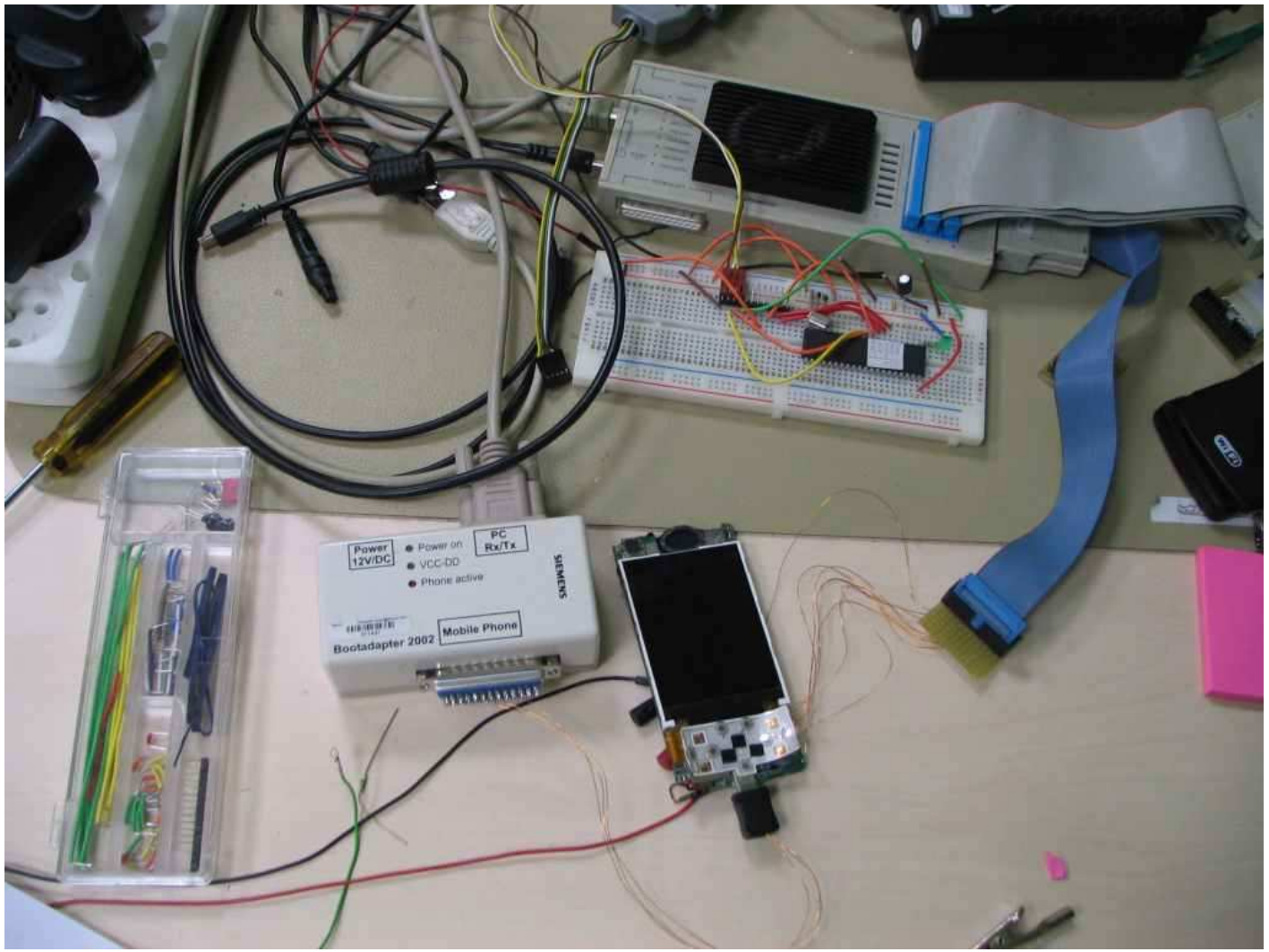
What do we need, to run Linux on new platform?

- Datasheet for the SoC
- Know how to run custom code
- Debug interface (JTAG, UART)
- Is cpu architecture already supported by Linux kernel?

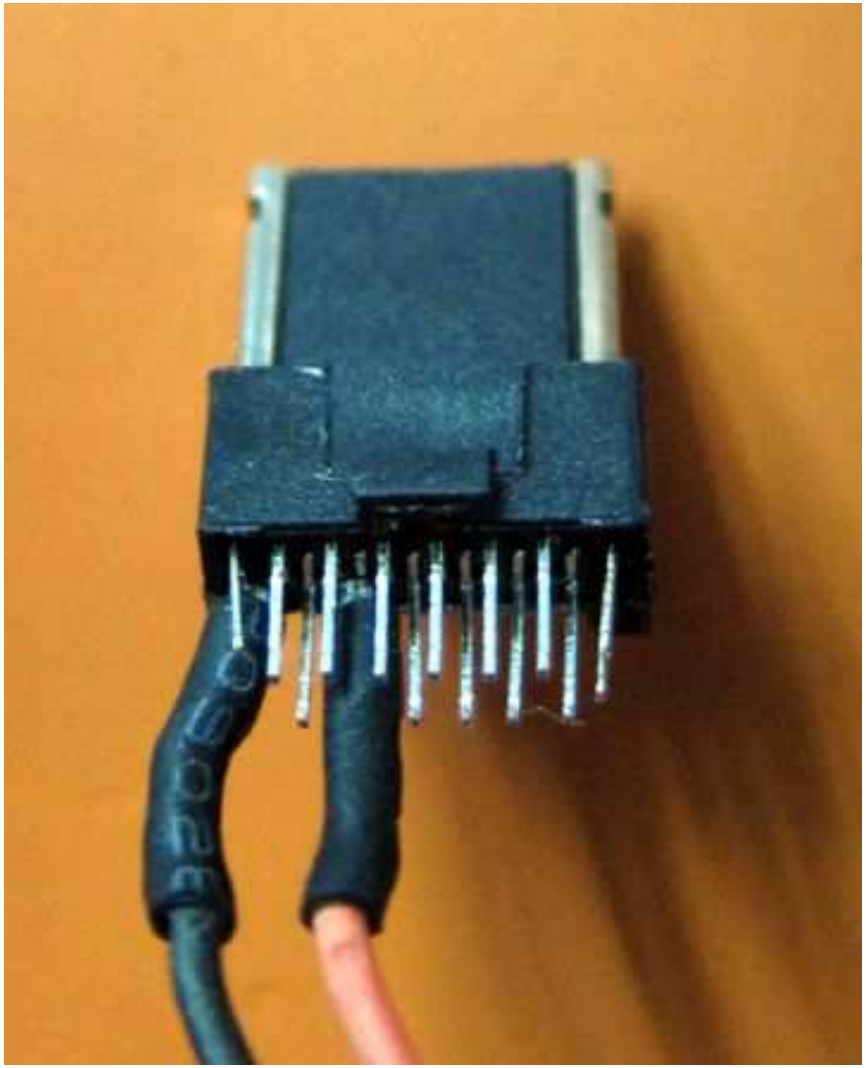
Finding HW pins

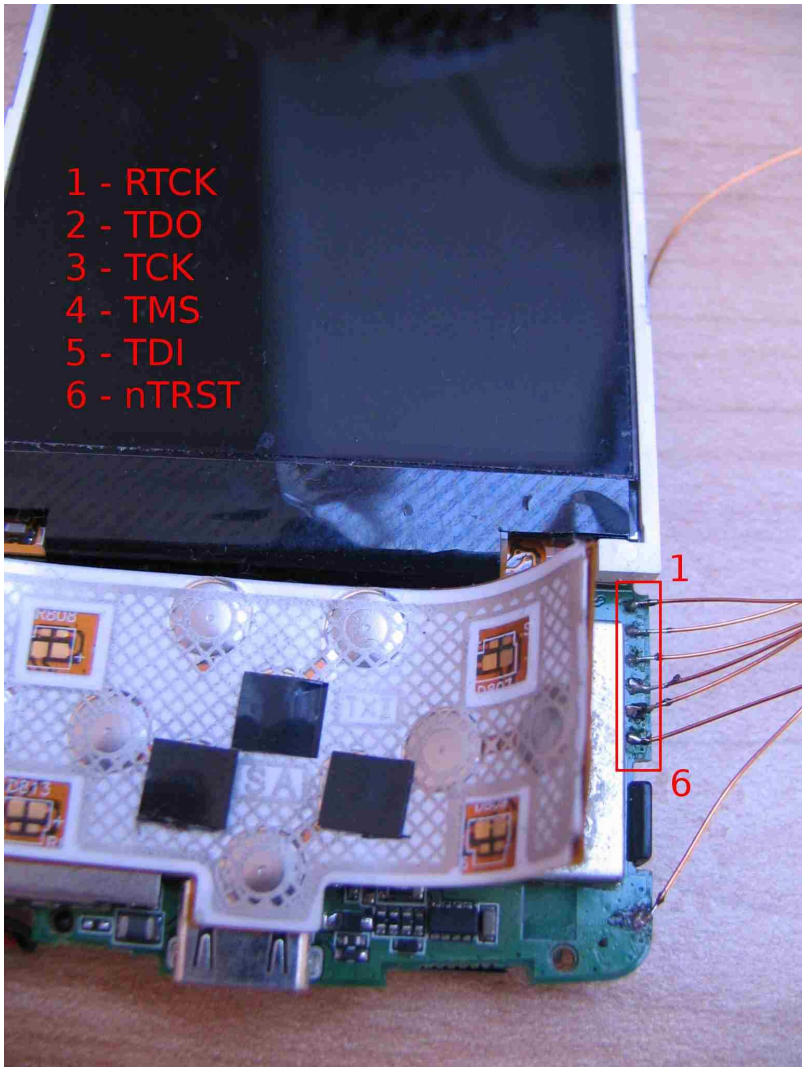
- We need to have interface to load binary code to mobile phone
- Finding JTAG will speed up development a lot, so it's worth to spend time on it
  - Very often not populated on PCB
  - At least 4 pins to find (TCK, TDO, TDI, TMS)
- UART is easier to find (just 2 pins) and very often available on external connector
- Use software for that, i.e. JTAG finder:
  - Built on ATMega32 (3.3V - 5V, 32 GPIOs)
  - Easy to build, even on solderless breadboard
  - Scanning of pins takes couple of seconds

JTAG finder hardware



JTAG and UART pins





Executing own code

- Try to find flashing tool for given SoC
  - Usually such tools upload loaders which are executed on target
  - Such loaders have code for specific peripherals (i.e. flash/RAM memory)
  - Loader can be signed
  - Sometimes you're able to load your own code using this tool
  - Start from sniffing communication between PC and target
- If JTAG has been found, much easier to analyze code
  - Direct access to registers, memory, peripherals
  - Easy to load code
  - Realtime debugging (current status of HW state)

SDRAM initialization

- First problem: How to init SDRAM memory?
  - Find out memory chip model and get datasheet
  - Disassemble loader uploaded by flasher (if loader contains SDRAM initialization)
  - Disassemble bootloader code
- MT6235 has 64kB static RAM, where SBL is loaded
- Even on the same model of phone, peripherals can differ (NAND, SDRAM, keypad, LCD)

Porting U-Boot

- Getting U-Boot running on new platform is extremely easy (if SoC is based on ARM)
- Just two drivers are needed to get U-Boot prompt:
  - UART
  - Timer
- Even if you see U-Boot source code for the first time it shouldn't take more than one day to get it running on new platform
- Bootloader is a good place to understand how peripherals work (testing basic drivers)

U-Boot UART driver

```
static void mt62xx_putc(int portnum, char c)
{
/* Wait until there is space in the FIFO */
while(!(readw(port[portnum] + MTK_UART_LSR) & UART_
WATCHDOG_RESET();

/* Send the character */
writew(c, port[portnum] + MTK_UART_DR);
}
```

U-Boot UART driver

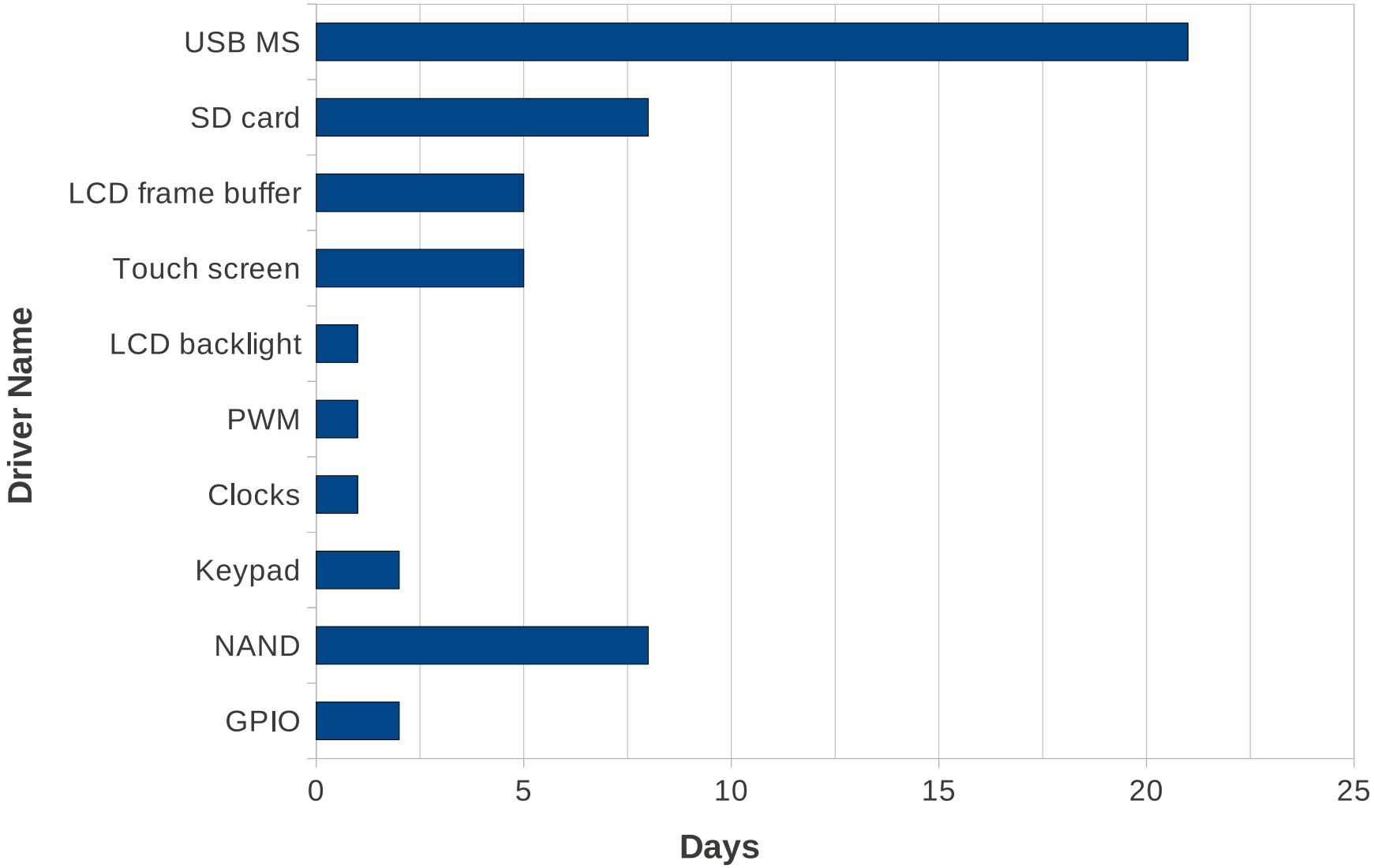
```
static int mt62xx_getc(int portnum)
{
/* Wait until there is data in the FIFO */
while (!(readl(port[portnum] + MTK_UART_LSR) & UART
WATCHDOG_RESET();

return readl(port[portnum] + MTK_UART_DR);
}
```

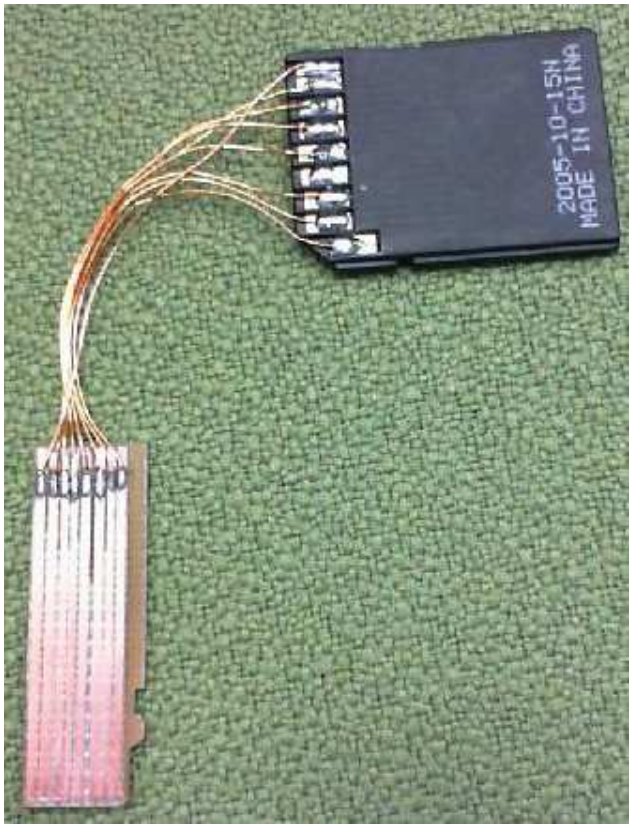
Porting Linux

- Assumption: Architecture is already supported (i.e. ARM)
- Linux porting of course takes longer than U-Boot porting
- To get Linux prompt following drivers are needed:
  - UART
  - Timer
  - Interrupt controller
- Add some constant definitions, generic functions and default configuration
- Usually it takes one week to get prompt in Linux

Timeline

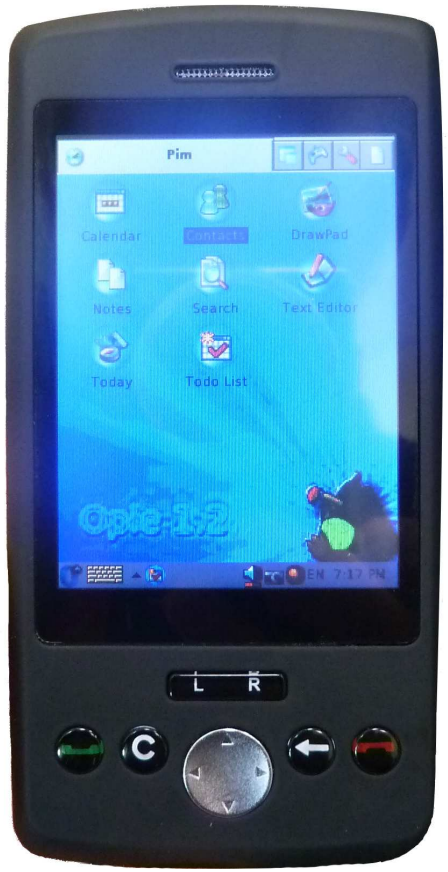

USB MS
SD card
LCD frame buffer
Touch screen
LCD backlight
PWM
Clocks
Keypad
NAND
GPIO
Driver Name
0
5
10
15
20
25
Days

Additional hardware

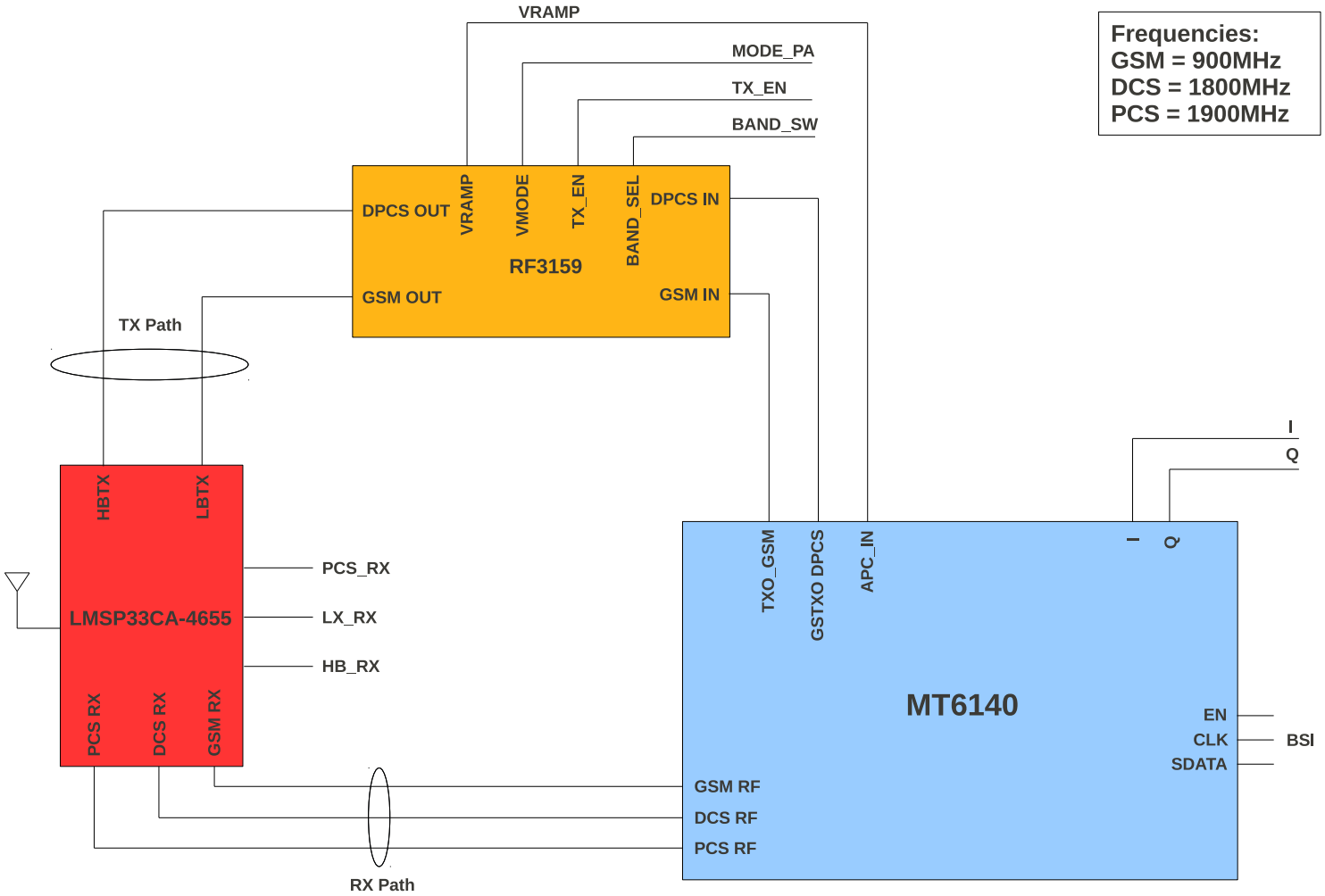


Running Linux distro

- OpenEmbedded used to build Linux distribution
- When drivers are already implemented it works out of the box
- OPIE (Open Palmtop Integrated Environment)
  - Graphical user interface for PDAs
  - A lot of applications and games available
  - Minimal requirements:
    - CPU: 80386, ARM 7
    - Touch screen 320x240
    - 10MB of flash memory
- It’s possible to run "real" Android 1.5

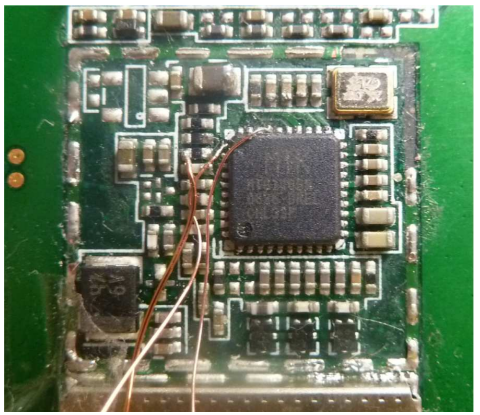


GSM RF simplified schematics

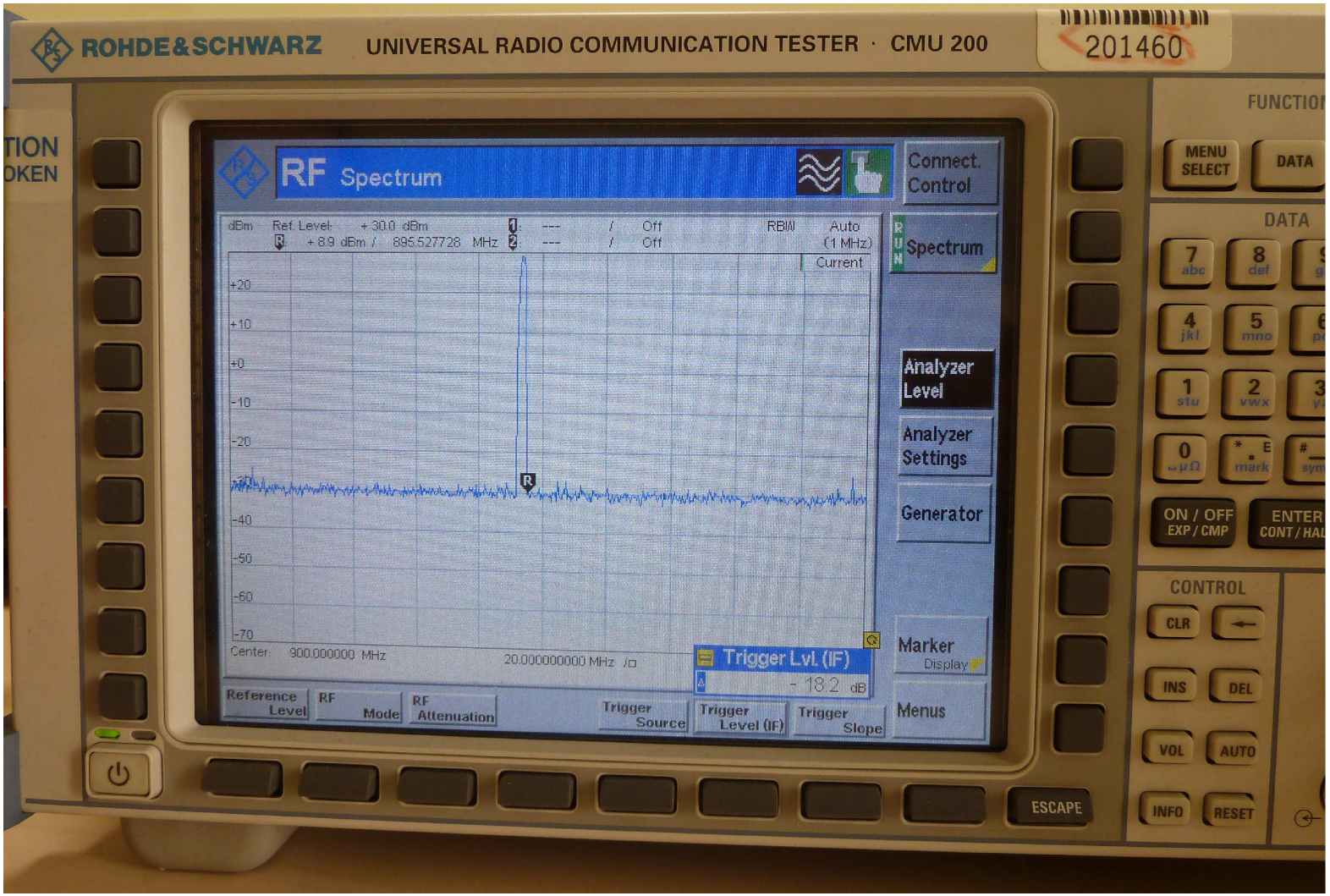


GSM RF chips

- Drivers written in U-Boot for following RF HW:
  - Murata LMSP33CA-465 - antenna switch
  - RF3159 - dual-mode amplifier
  - MT6140 - GSM/GPRS/EDGE RF transceiver
  - BSI - Baseband Serial Interface
  - BPI - Baseband Parallel Interface
  - BFE - Baseband Front End
  - TDMA - Time Division Multiple Access
  - APC - Automatic Power Control
- U-Boot command:
  - rf_tx <arfcn>



Testing drivers for TX path



DSP reverse engineering

- DSP in BaseBand ASIC is the biggest secret of manufacturers
- In MT6235 datasheet there are 20 pages missing (in ARM-DSP interface chapter)
- Very often this part has no documentation at all

DSP reverse engineering

Facts about DSP in MT6235:

- Most probably Analog Devices ADSP-2181
- Code for DSP is located in ROM (not downloaded over IDMA)
- DSP patch unit exists (possibility of potential hack)
- So far we didn’t manage to execute own code on DSP
- So far we didn’t manage to dump existing code on DSP
- Best approach would be to use DSP as black box

Yet to do:

- Investigate more on DSP (blocking point at the moment)
- Port OsmocomBB to MTK HW
- Adopt OsmocomBB to Linux

Final result:

- We can get first fully open source mobile phone
- Lots of possibilities (acquiring logs, sniffing, etc.)

Summary

- Fake mobile phones market is really interesting
- Porting Linux to new platform based on ARM is easy
- HW/SW reverse engineering takes long time
- Running own code is most important step
- Baseband chips are well protected by manufacturers
- We're on good track to get first fully open source mobile phone

Does Mediatek platform have any future?

- MT6516
  - ARM926EJ-S @ 416MHz (application)
  - ARM7 @ 104MHz (baseband)
  - Quad-Band GSM/GPRS/EDGE
- MT6573 (1GHz = 650MHz + 400MHz)
  - ARM11 @ 650MHz (application)
  - ARM9 @ 400MHz (baseband)
  - WCDMA + GSM (MT6162)
- A lot of peripherals from new SoCs are very similar to MT6235
- Tons of MTK based phones are cheap and available on the market
- MTK hardware is very stable!

Touch Galaxy S2 HDC A9100/i9100 1GHz Android

Questions?