Make your own M2M application, in ½ hour, with Lua

Fabien Fleutot

Lua Workshop ’12

Sierra Wireless @ Lua workshop

Last year, we talked about our Lua based M2M SDK
http://www.lua.org/wshop11/m2m-embedded-development-with-lua.pdf

Today, we'll actually demonstrate it; sources available on github: http://github.com/fab13n/wshop12

But we'll inflict you some more talking first, for those who missed or forgot last year's talk!

We promised to open source it, we partially did:
https://github.com/SierraWireless/luasched

We'll do more: http://www.eclipse.org/mihini

What’s M2M?

M2M == Machine-to-Machine [communications]

(The latest fashionable name for M2M is “IoT”, i.e. “Internet of Things”)

M2M is networking for embedded devices, with some twists:

- Hardware is disseminated over vast areas
- Access primarily happens through GSM / CDMA / 3G networks
- There's no skilled operator / maintainer on site
- Fleets are often large and heterogeneous

About Sierra Wireless

We design and build M2M modems

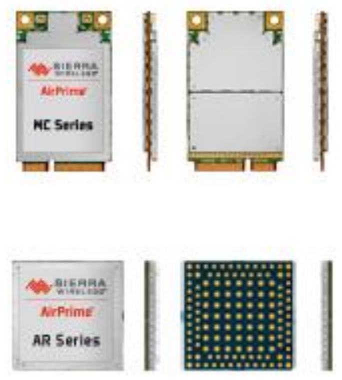
Our customers build solutions with them

We want more solutions, built faster and for cheaper

SIERRA WIRELESS™

About Sierra Wireless

We design and build M2M modems



Our customers build solutions with them

We want more solutions, built faster and for cheaper

About Sierra Wireless

We design and build M2M modems

Our customers build solutions with them:

- Energy (windmill, solar panels)
- Dispensers: ATM, vending machines, charging stations
- Specialized engines: compressors, water plants, coffee machines
- Mobile assets: vehicle fleets, shipping containers
- Utility meters

We want more solutions, built faster and for cheaper

About Sierra Wireless

We want more solutions, built faster and for cheaper

- A solution involves:
  - Embedded hardware
  - Embedded Software
  - IP Networking
  - Wireless networking
  - Radio issues
  - Protocols
  - Back-end server software
  - Database
  - Front-end server software
  - ERP integration
  - Telecom operators integration

About Sierra Wireless

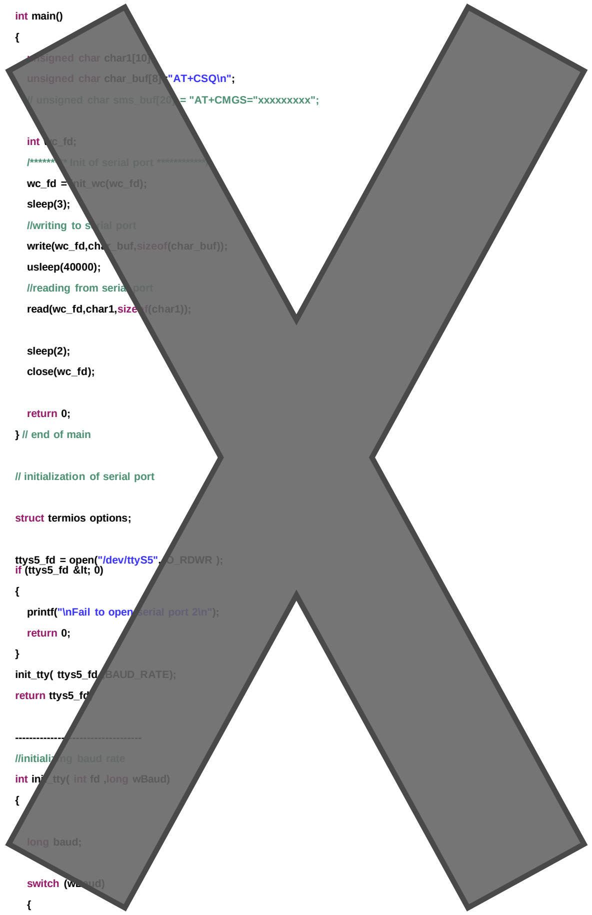
We want more solutions, built faster and for cheaper

- A solution involves *[many domains].*
- None of this is rocket science, but very few organizations are competent in all of these domains simultaneously.

Billions of M2M devices are forecast in the next decade

- That’s quite a bubble
- It cannot be only manned by embedded specialists.
  The market will belong to those who enable generalist developers.

“Embedded expertize”



```
int main()
{
   unsigned char char1[10];
   unsigned char char_buf[8] ="AT+CSQ\n";
   // unsigned char sms_buf[20] = "AT+CMGS="xxxxxxxxx";

   int wc_fd;
   /********* Init of serial port ************/
   wc_fd = init_wc(wc_fd);
   sleep(3);
   //writing to serial port
   write(wc_fd,char_buf,sizeof(char_buf));
   usleep(40000);
   //reading from serial port
   read(wc_fd,char1,sizeof(char1));

   sleep(2);
   close(wc_fd);

   return 0;
} // end of main

// initialization of serial port

struct termios options;

ttys5_fd = open("/dev/ttyS5", O_RDWR );
if (ttys5_fd &lt; 0)
{
   printf("\nFail to open serial port 2\n");
   return 0;
}
init_tty( ttys5_fd ,BAUD_RATE);
return ttys5_fd;

-----------------------------------
//initializing baud rate
int init_tty( int fd ,long wBaud)
{

   long baud;

   switch (wBaud)
   {
```

```
sms.send(
  '+33612345678’,
  'My SMS’,
)
```

It’s not OK for a simple, core operation such as sending an SMS to take pages of error-prone code. Yet it’s still the norm in the embedded world.

About Sierra Wireless

We want more solutions, built faster and for cheaper

- A solution involves *[many domains]*
  […]
- The market will belong to those who enable generalist developers.

Sierra Wireless provides:

- Embedded hardware
- SIM / Subscription / Airtime billing management
- Embedded SDK (in Lua): API, runtime, IDE
- Back-end servers
- REST access to servers
- Generic UI, designed for large fleet management

Demo time

DIY-friendly hardware:

- Raspberry Pi
- Arduino Uno (representative of distributed architectures)
- Plugs Arduino shield (to avoid any soldering)
- Cheap, generic sensors, wired straight to GPIOs

<$100, available on the net, accessible to all developers.

Enables fun projects: domotics, automated greenhouse, RC models…

Demo time

1. Simplest possible application: telnet server + scheduler
2. Getting data: modbus
3. Making data physically meaningful
4. Publishing with MQTT
5. Controlling through http://m2mop.net

Sources: http://www.github.com/fab13n/wshop12

API: http://developer.sierrawireless.com/en/Resources/Resources/AirLink/ALEOS_AF/RefDoc_ALEOS_AF_API.aspx

[Demo: basic modbus, data processing, MQTT connection]

Embedded Agent services

Embedded devices need to perform:

- Data acquisition / consolidation / reporting
- Locally managed actions
- Server-initiated actions
- Over-The-Air software and firmware updates

Embedded Agent services

Embedded devices need to perform:

- Data acquisition / consolidation / reporting
  - **access to local I/O: serial buses, GPIO, ADC/DAC, LAN…**
  - **local storage**
  - **optionally persisted**
  - **efficient encoding of time series**
  - **customizable precision/bandwidth compromize**
  - **standard consolidation methods (min, max, mean...)**
  - **remotely customizable reporting policies**
- Locally managed actions
- Server-initiated actions
- Over-The-Air software and firmware updates

Embedded Agent services

Embedded devices need to perform:

- Data acquisition / consolidation / reporting
- Locally managed actions
  - **get / set / notify API for system state**
  - **full Lua programming language, with I/O APIs**
- Server-initiated actions
- Over-The-Air software and firmware updates

Embedded Agent services

Embedded devices need to perform:

- Data acquisition / consolidation / reporting
- Locally managed actions
- Server-initiated actions
  - **get / set / notify API for server-controlled data**
  - **standard encoding of commands and handlers**
  - **can be sent to / acknowledged by large batches of devices**
- Over-The-Air software and firmware updates

Embedded Agent services

Embedded devices need to perform:

- Data acquisition / consolidation / reporting
- Locally managed actions
- Server-initiated actions
- Over-The-Air software and firmware updates
  - **Firmware and software authentication**
  - **Management by arbitrary batches**
  - **Monitoring of success / failure**
  - **Auto-recovery in case of failure**
  - **Update forwarding to other embedded CPUs**

[Demo: Embedded Agent+m2mop.net, reporting, setting, command]