

Node-level Representation and System Support for Network Programming

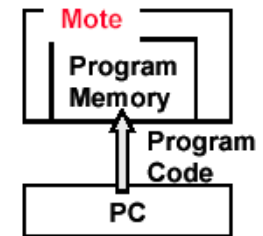
Jaein Jeong

Intro – Direct Connection Programming

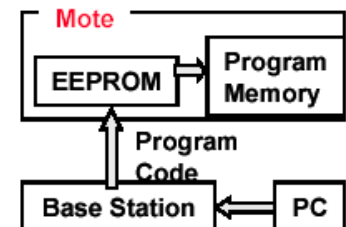
- Steps for programming wireless sensors
 - The code is developed in a host machine (e.g. PC).
 - And the code is loaded to a single sensor node, typically, with direct connection like parallel cable.
- Problems with the direct connection programming
 - Programming time increases with # of sensor nodes.
 - Involves all the efforts of collecting the sensor nodes placed in different locations and possibly disassembling and reassembling the enclosures.
- Network reprogramming can address scalability and reassembly issues.

Intro – Network Reprogramming.

- Disseminates program code to one or more sensor nodes through radio channel.
- Network Reprogramming Steps
 - Program code is stored in external flash.
 - Possible retransmission of lost packets.
 - The boot loader copies the program code from external flash to program memory. Program starts after reboot.
- Supported in TinyOS 1.1, but not optimized for fast delivery.
 - Sends all the program code even though the changes from the previous version are small.



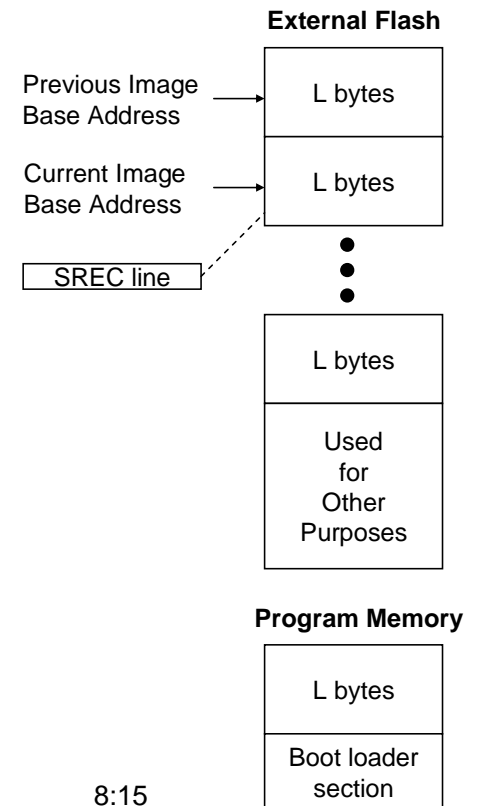
Parallel programming.



Network programming

Design - Memory Layout

- The program storage (external flash) is organized in multiple sections. For now, each sensor maintains two sections: current and previous sections.
- In each section, each line of the program binary (SREC file) is stored as a 32-byte record.
- Boot loader is modified so that it can copy code from any section on external flash rather than a single location.

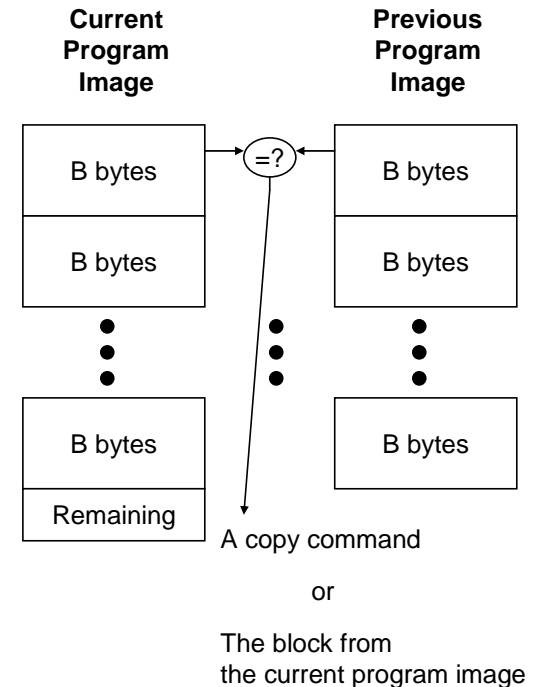


SREC line

0:1	2:3	4	5	6:7	8:15
PID	CID	Srec Type	length	Srec address	Data
16: 8+datalen-1				8+datalen	Remaining
Data				Checksum	Fill

Design - Generating update

- We assume that a host machine can keep the program history of the sensor nodes.
- To generate program update, we compare the program image in fixed size blocks.
 - The blocks (256 bytes) from the current and the previous images are compared sequentially.
 - If they match, “copy” command is sent which makes the nodes to copy the block from the previous image to the current one.
 - Otherwise, the block from the current program image is sent.



Experiments

- Case 1 (Original Xnp)
 - Original network reprogramming with XnpCount app.
- Case 2 (Best case)
 - Sending the update for two copies of XnpCount (just timestamps are different.)
- Case 3
 - Sending the update from XnpCount to XnpRfmToLeds.

Three scenarios	Case 1	Case 2	Case 2
# lines in SREC file	1166 lines	1166	1145
# bytes in SREC file	18.6 KB	18.6 KB	18.3 KB
Time to complete	223 sec	75 sec	175 sec
Effective BW	83.4 bytes/sec	248 bytes/sec	104.5 bytes/sec

Discussion & Future works

- Not much of code is shared in binary files.
 - Two scenarios are considered to test the level of sharing the binary code.
 - Considering that the change is small compared to the whole program size, this is far from the optimal.
 - Needs better compiler support for maximizing binary code reuse.

Two scenarios	Case 1 (With different timestamp)	Case 2 (Additional lines were added.)
# lines in SREC file (old)	1027	1027
# lines in SREC file (new)	1027	1032
# lines to transmit	1	635
# lines to copy	1026	397

Discussion & Future works

- Organizing program storage in directory structure.
 - Currently, a series of contiguous blocks are allocated for the current and the last program image.
 - Sharing blocks in a single pool has some advantages:
 - Economical use of memory: blocks of the same content are shared among different images.
 - Avoids external flash operation overhead by keeping pointer rather than copying a block of lines.
 - A user doesn't need to keep track of program history of sensor nodes.
 - The directory structure requires major change of network reprogramming modules.
- Modification for multi-hop delivery.