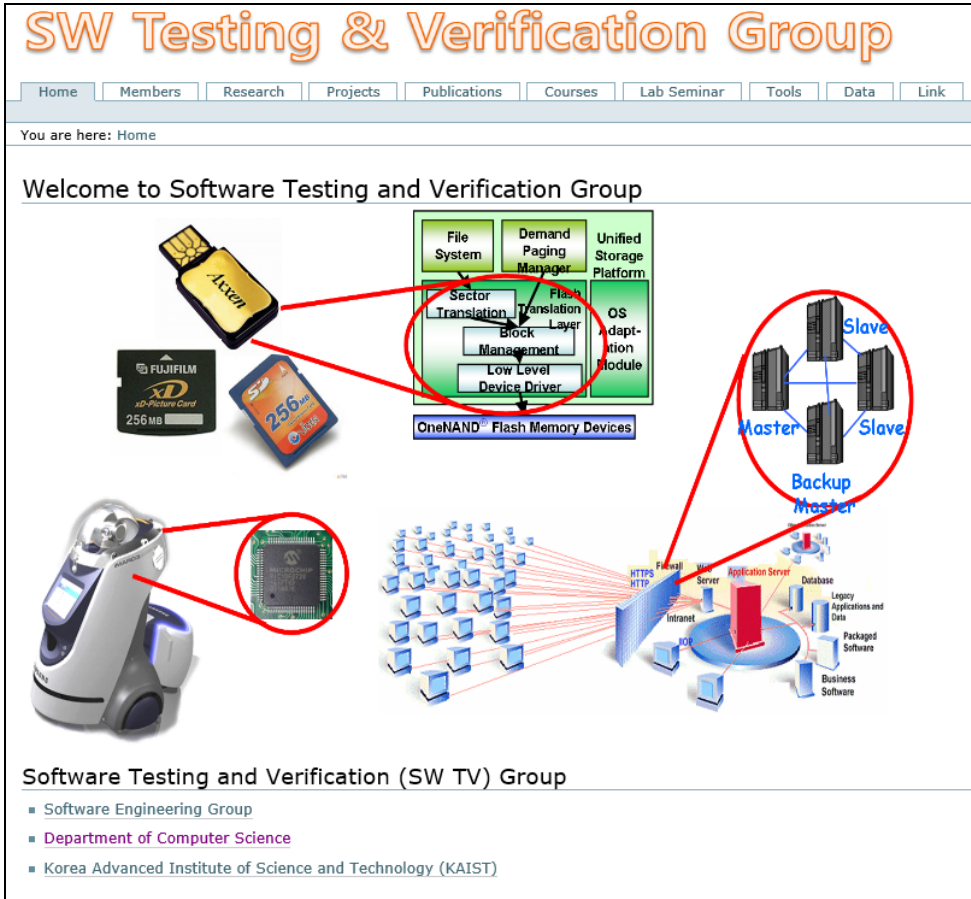
CS453: Automated Software Testing

Moonzoo Kim

Software Testing and Verification Group

CS Dept. KAIST

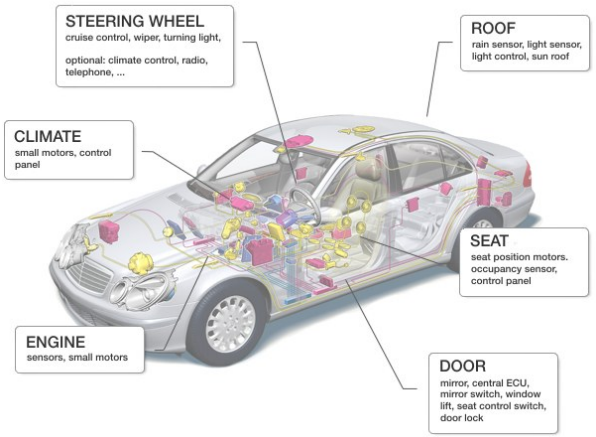


Strong IT Industry in South Korea

SAMSUNG

LG

KIA MOTORS



HYUNDAI

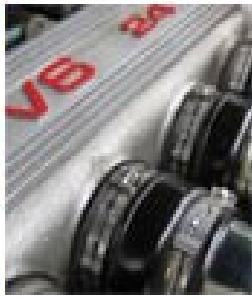
Embedded Software in Two Different Domains

Conventional Testing

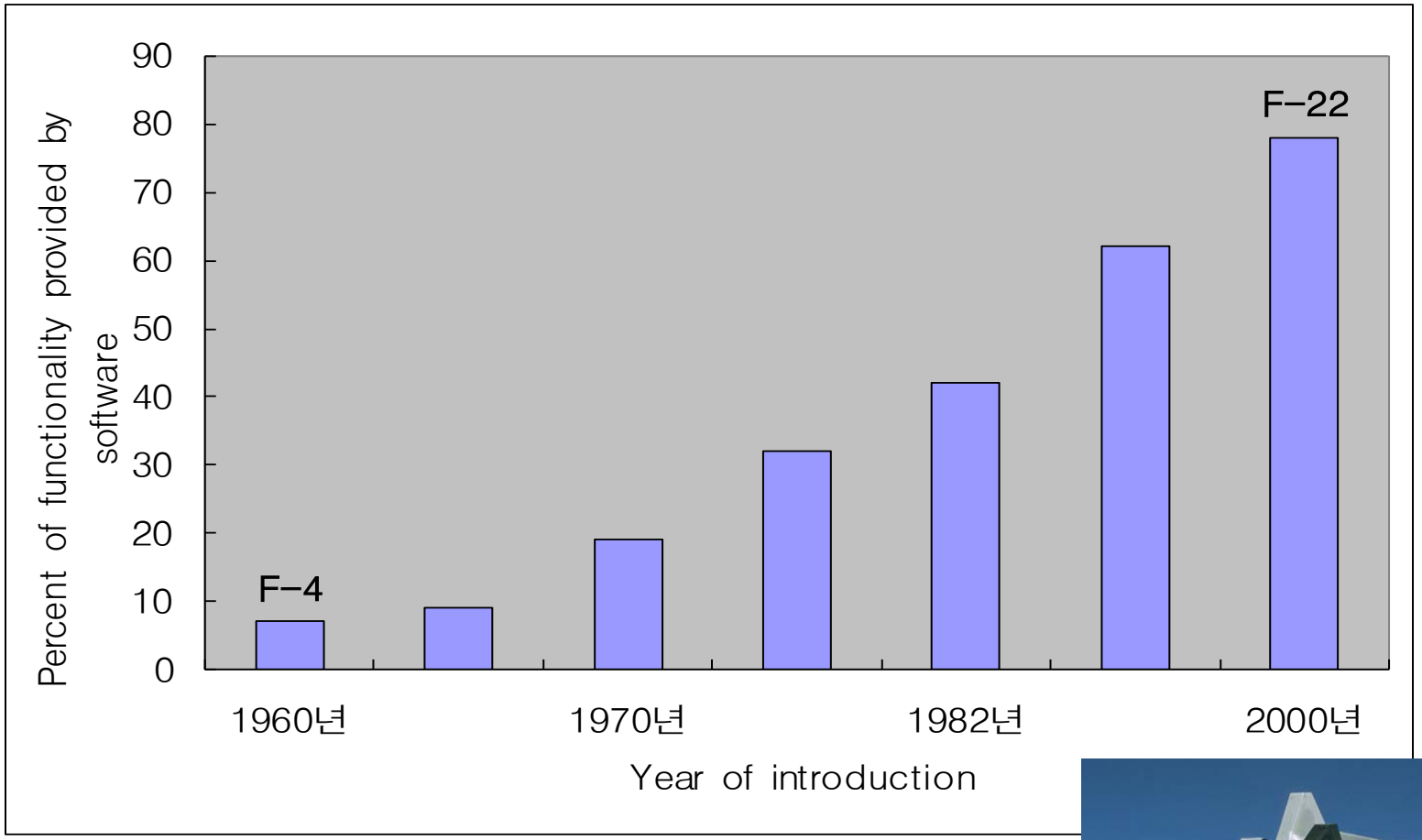
Concolic testing

Model checking

	Consumer Electronics	Safety Critical Systems
Examples	Smartphones, flash memory platforms	Nuclear reactors, avionics, cars
Market competition	High	Low
Life cycle	Short	Long
Development time	Short	Long
Model-based development	None	Yes
Important value	Time-to-market	Safety

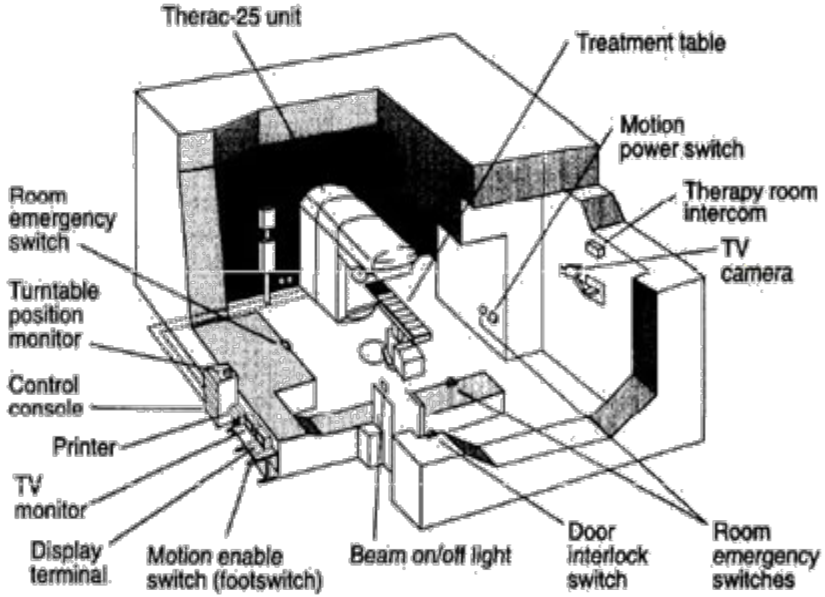


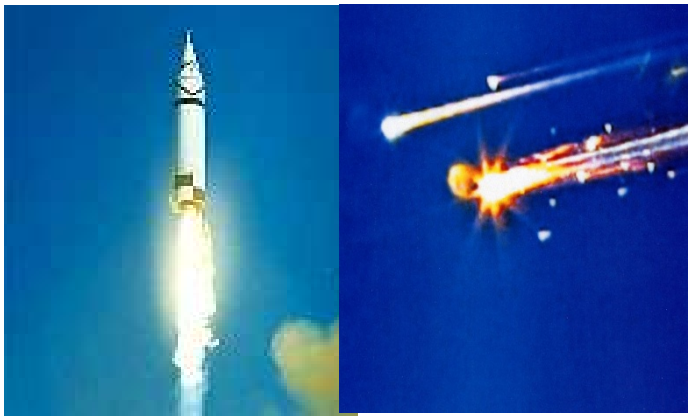
Role of S/W: Increased in Everywhere

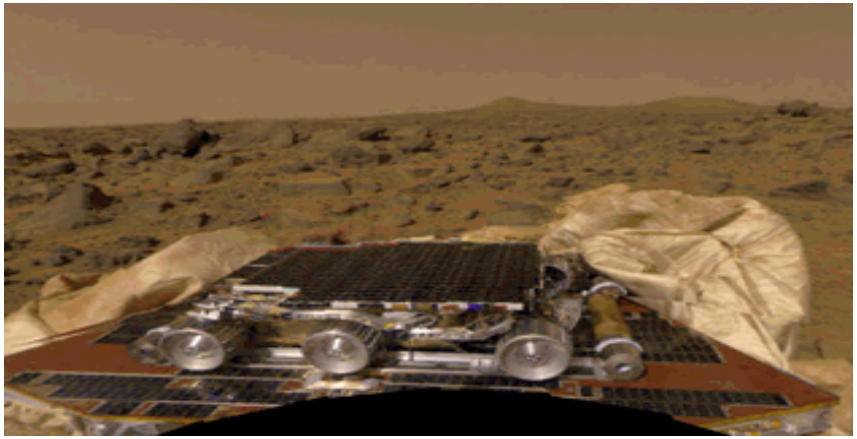


자료출처: Watts Humphrey 2002

Motivation: Poor Quality of SW





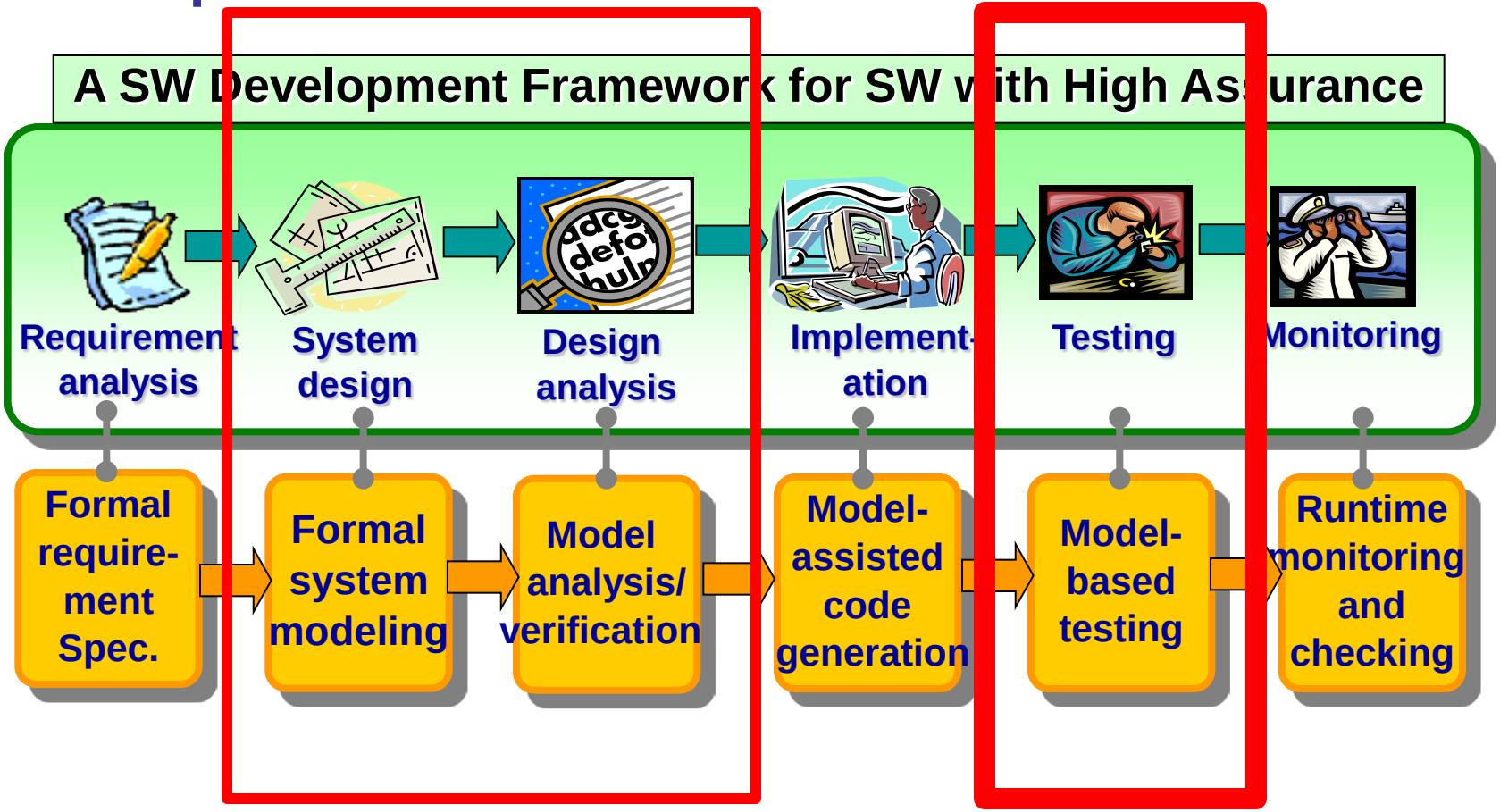


Current Practice for SW

- **SW developers have to follow systematic disciplines for building and analyzing software with high quality**
  - This class focuses on the analysis activities

Software Development Cycle

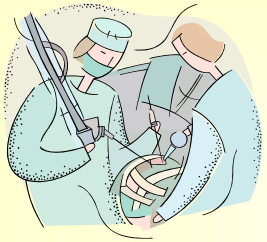
- A practical end-to-end formal framework for software development



Main Target Systems

- **Embedded systems where highly reliable SW technology is a key to the success**
  - The portion of SW in commercial embedded devices increases continuously
  - More than 50% of development time is spent on SW testing and debugging

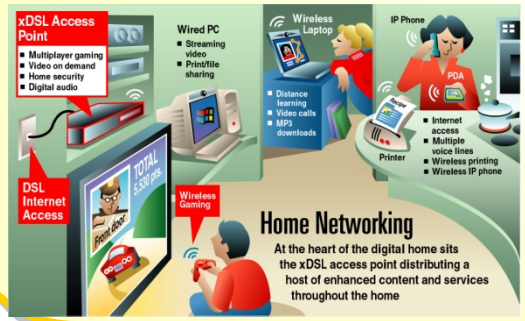
Intelligent Medical Devices



Home Service Robots

Highly Reliable Systems

Home Network



Intelligent Mobile Systems

How to Improve the Quality of SW

1. **Systematic testing (can be still manual)**
   - Coverage criteria
   - Mutation analysis

2. **Testing through automated analysis tools**
   - Scientific treatment of SW with computing power
   - Useful tools are available

3. **Formal verification**
   - Guarantee the absence of bugs

Questions???

- **Is automated testing really beneficial in industry?**
  - Yes, dozens of success stories at Samsung
- **Is automated testing academically significant?**
  - Yes, 3 Turing awardees in ‘07
- **Is automated testing too hard to learn and use?**
  - No, there are tools available

Companies Working on Software Verification

Microsoft

SAMSUNG

INRIA

cadence

IBM

ESTEREL Technologies

Sparrow The Early Bird

NEC Empowered by Innovation

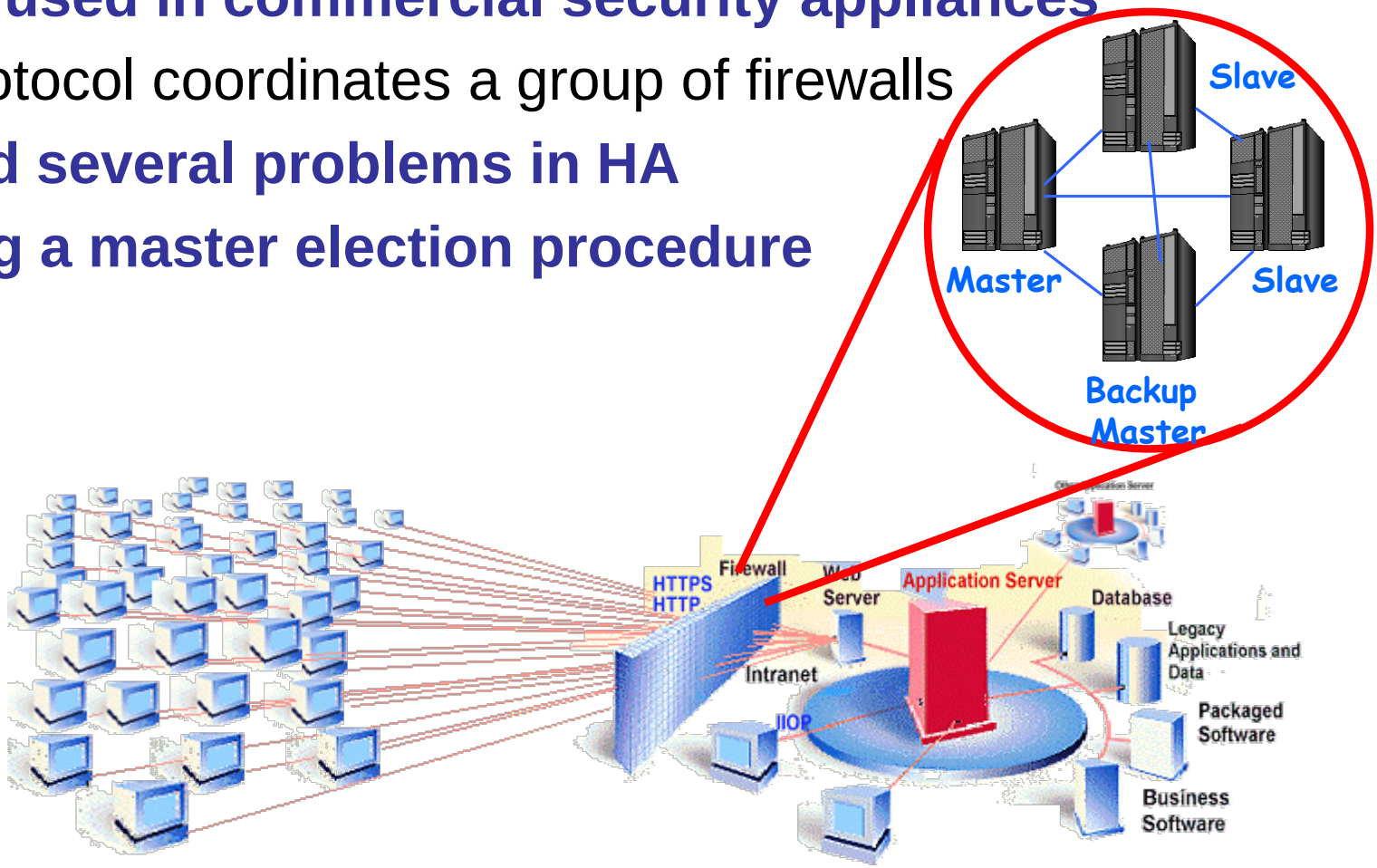
Sun microsystems

NASA Jet Propulsion Laboratory California Institute of Technology

The MathWorks™ Accelerating the pace of engineering and science

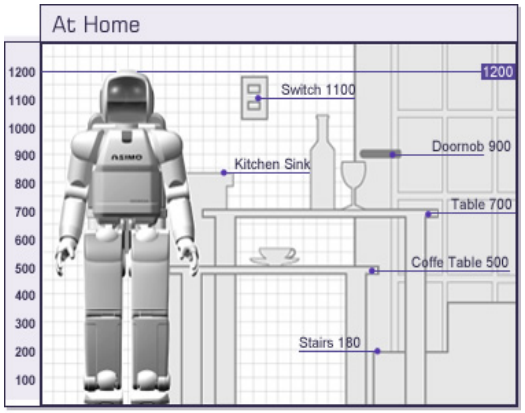
Verification of High-Availability Protocol

- **We develop a formal model of high-availability protocol used in commercial security appliances**
  - HA protocol coordinates a group of firewalls
- **We found several problems in HA regarding a master election procedure**

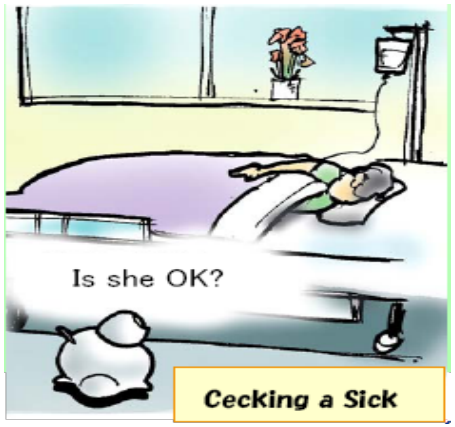


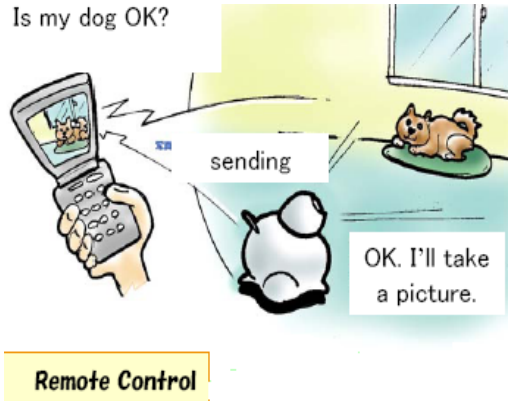
Home Service Robot

- Designed for providing various services to human user
  - Service areas : home security, patient caring, cleaning, etc



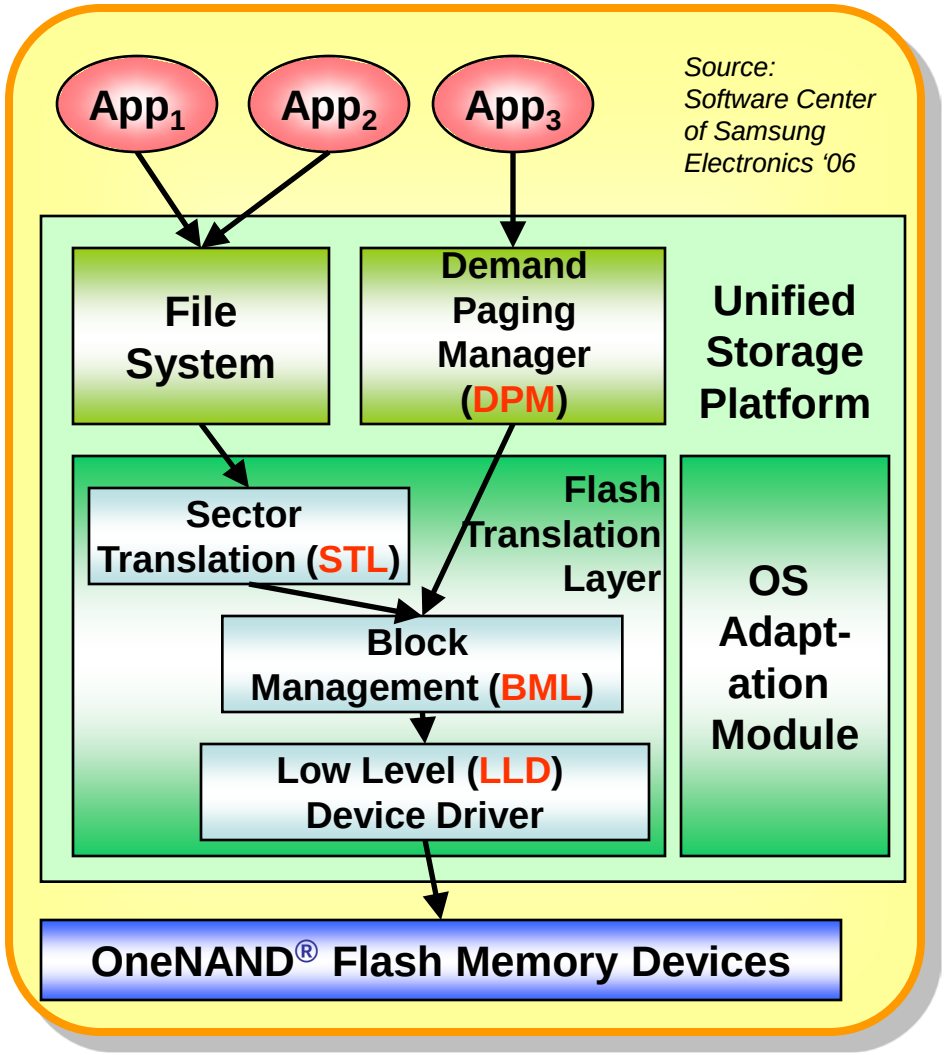
* The above heights are examples to serve as a reference(mm).





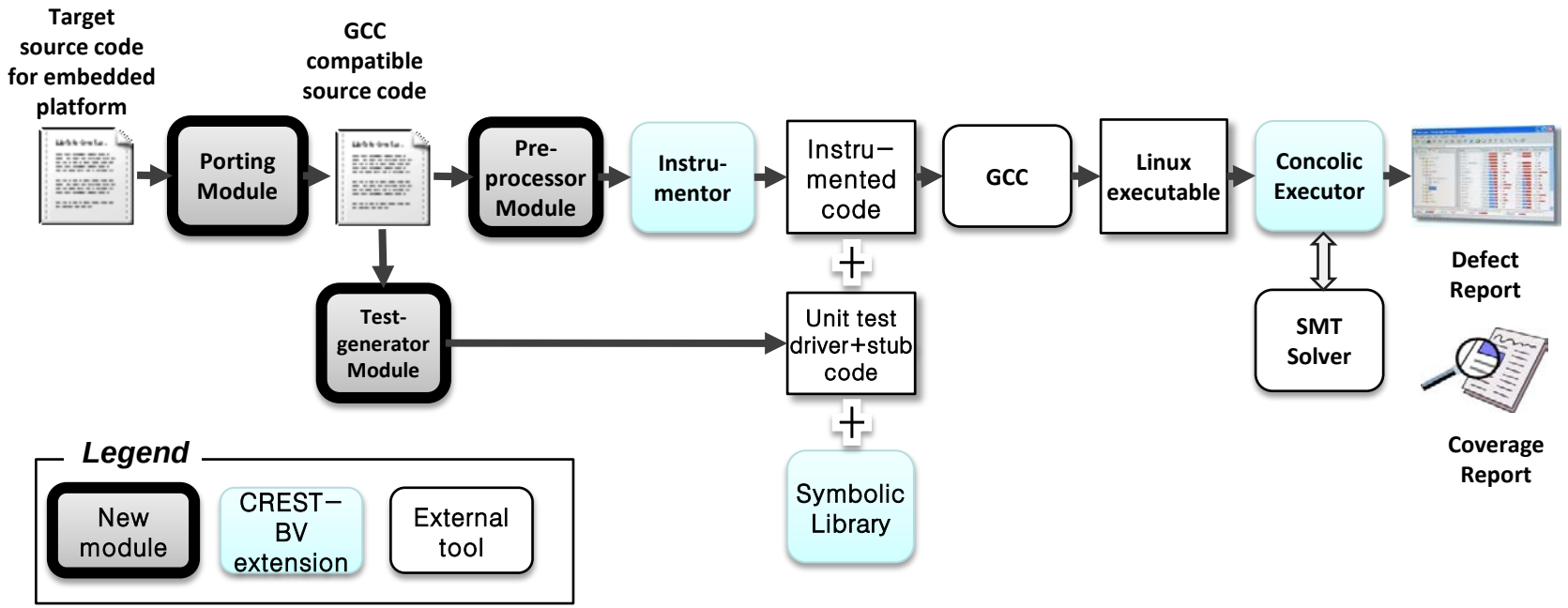
OneNAND® Flash Memory

- Each memory cell can be written limited number of times only
- XIP by emulating NOR interface through demand-paging scheme
- Performance enhancement



Smartphone SW Platform

- We have developed CONcrete and symBOLic (CONBOL) framework that is an automated concolic unit testing tool based-on CREST-BV for embedded software



Research Trends toward Quality Systems

- **Academic research on developing embedded systems has reached stable stage**
  - just adding a new function to a target system is not considered as an academic contribution anymore
- **Research focus has moved on to the quality of the systems from the mere functionalities of the systems**
  - Energy efficient design, ez-maintenance, dynamic configuration, etc
- **Software reliability is one of the highly pursued qualities**
  - ASPLOS 2011 Best paper
    - "S2E: a platform for in-vivo multi-path analysis for software systems" @ EPFL
  - OSDI 2008 Best paper
    - "Klee: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs" @ Stanford
  - NSDI 2007 Best paper
    - "Life, Death, and the Critical Transition: Finding Liveness Bugs in Systems Code" @ U.C. San Diego

Tool-based Interactive Learning

- **Model checker**
  - Explicit model checker: Spin home page
  - Symbolic model checker: NuSMV home page
- **Software model checker**
  - Bounded model checker for C program:CBMC home page
  - Predicate abstraction for C program: BLAST home page
- **Satisfiability solver**
  - **MiniSAT home page**
- **Satisfiability Module Solver**
  - **Yices home page**
  - **Z3 home page**
- **Concolic testing tools**
  - **CREST home page**

Class Schedule

- wk1: overview on automated SW analysis techniques
- Wk2-3: coverage based SW
- wk4: background on Propositional logic and SAT (Satisfiability) solvers
- wk5: SAT solver heuristic and tool application 1: MiniSAT
- wk6: background on First order logic
- wk7: Satisfiability Modulo Theory (SMT) basic
- wk8: midterm exam
- wk9: advanced application of SMT solvers
- wk10: directed automated rand om testing
- wk11: tool application : CREST
- wk12: basic temporal logic for r equirement property
- Wk13-14: tool application: Spin & NuSMV
- wk15: state space minimization techniques
- wk16: final exam

Final Remarks

- **For undergraduate students:**
  - Highly recommend URP studies or independent studies
    - Ex. 이준희 (05학번) got a silver award and macbook air notebook ☺
      - Debugging Linux kernel through model checking to detect concurrency bugs
    - Ex2. Nam Dang wrote down a paper on distributed concolic testing
      - Y.Kim, M.Kim, N.Dang, Scalable Distributed Concolic Testing: a Case Study on a Flash Storage Platform, Verified Software Track @ Intl. Conf. on Theoretical Aspects of Computing (ICTAC), Aug 2010

Final Remarks

- **For graduate students:**
  - Welcome research discussions to apply formal analysis techniques
    - Systematically testing/debugging C programs
    - Concurrency bug detection
    - Model-based testing
- **Pre-requisite:**
  - Basic understanding of the C programing language
  - Basic understanding of linux/unix