전력 소모 최소화를 통한 성능 개선의 코드 가시화 방법

안현식*, 박보경*, 김영철**
*홍익대학교 일반대학원 소프트웨어공학연구실
**홍익대학교 소프트웨어융합학과
{ahn*, park*}@selab.hongik.ac.kr, bob**@hongik.ac.kr

Code visualization approach for performance improvement via minimizing power dissipation

Hyun Sik An*, Bokyung Park*, R.Young Chul Kim**
*,**SELab. Hongik University

요 약

높은 사양이 필요한 하드웨어 기반의 모바일 및 IoT 임베디드 시스템은 저전력과 성능에 중요한 이슈를 갖고 있다. 이는 전력 소비로 발열량 증가 및 기기의 수명 단축 문제가 발생된다. 이러한 환경에서 소프트웨어도 제한된 전력, 메모리 등에서 안정적인 동작을 수행해야 하므로 디바이스의 소비전력이 증가한다. 이를 해결하고자, 코드 관점에서 전력 소모 최소화를 통한 소프트웨어 성능 개선 가시화 방법을 제안한다. 이는 코드 가시화를 통해 복잡한 모듈을 식별하고, 저전력 코드 패턴을 적용하여 소프트웨어 성능을 개선한다. 이런 코드로 소비전력을 감소 및 성능을 개선함으로써 코드의 품질을 최적화 할 수 있다.

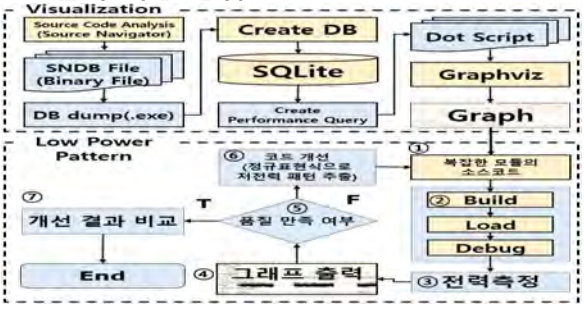
1. 서론

최근 자율주행, 드론, 스마트 폰 등 다양한 분야에서 임베디드 기기의 역할과 기능이 점차 확대되고 있다. 임베디드 시스템은 높은 사양을 가진 하드웨어로 구성된다. 고사양의 하드웨어에서 고성능의 소프트웨어가 작동함으로써 소비전력이 증가하고 있으며, 임베디드 시스템의 소비전력 증가는 기기의 수명단축으로 이어진다. 이러한 문제 해결을 위해 본 논문에서 소프트웨어 코드의 전력 측정을 통한 전력 소모 최소화 및 성능 개선 방법을 제안한다. 이 방법은 소프트웨어 가시화를 통해서 코드의 복잡도를 추출한다. 추출된 복잡도 중에서 소프트웨어의 성능을 저하시키는 모듈을 선정한다. 선택된 모듈은 본 연구에서 새롭게 정의한 저전력 패턴을 적용하여 소비 전력을 개선하고, 가시화를 통해 성능이 개선됐는지 확인한다. 본 논문의 구성은 다음과 같다. 2장은 관련 연구를 소개한다. 3장은 전력 측정 실험을 통해 새롭게 정의한 저전력 코드 패턴을 언급한다. 4장은 사례연구, 5장은 결론 및 향후 연구를 기술한다.

2. 관련 연구

Vetro가 제안한 Energy Code Smell은 소모 전력을 감소시킬 수 있는 가능성이 높은 패턴이다[1]. 하지만 에너지의 소모가 증가되는 경우도 있어 소비 전력을 절감하는데 적합하다고 볼 수 없다. 기존 연구에서는 Loop문에 대해서만 논하였다[2]. 이를 보완 및 확장하는 패턴은 3장에서 언급한다. 소프트웨어 가시화는 비가시성인 소프트웨어를 가시화하는 기법을 의미한다. (그림 1)의 Visualization 영역은 본 연구에서 사용한 성능 가시화 흐름도이다. 먼저 Source Navigator를 통해 소프트웨어 내부 정보를 SNDB 파일로 추출한다. 이 파일을 DBdump를 이용하여 텍스트로 변환한다. 그 후 변환된 데이터를 데이터베이스 생성 후 각 테이블에 저장한다. 성능지표 추출을 위해 쿼리문을 생성한다. 결과물들을 Graphviz에 Dot Script로 입력하면 가시화 그래프에서 소프트웨어의 성능을 확인 할 수 있다[3].

3. 소프트웨어 코드의 전력 소모 최소화를 위한 소스코드 전력 측정 방법



(그림 1) 성능가시화 및 전력소모 최소화 절차 흐름도

(그림 1)은 전력 소모 최소화를 통한 성능 개선의 코드 가시화 방법의 흐름도이다. 가시화한 그래프의 모듈 중에서 소프트웨어의 성능을 저하시키는 모듈을 선정한다. (그림 1)의 Low Power Pattern 영역은 성능가시화를 통해 선정된 복잡한 모듈의 전력 측정을 통한 전력소모 최소화 방안이다. 먼저 소스 코드 전력측정을 위한 환경을 구성한다. ①Keil uVision5 IDE에서 측정하고자 하는 소스 코드를 입력한다. ②소스코드를 실행한다. 프로젝트 실행은 3단계로 구성되어있다. 입력한 소스코드를 Build하고 오류가 없으면 Load하여 보드에 Download한다. 그 후 Debug모드를 실행한다. ③디버깅 모드에 들어가면 입력된 소스 코드의 전력을 측정할 수 있다. 측정결과는 ④와 같은 그래프로 나온다. ⑤측정된 전력값과 그래프를 참고하여 소스코드의 품질 만족 여부를 결정한다. ⑥만약 만족하지 못한다면

전력소모를 최소화 하기 위한 코드로 개선한다. 만족한다면 소스 코드 전력 측정을 통한 전력소모 최소화 절차를 종료한다. (그림 2)는 측정 그래프다. ①번은 프로그램의 시작점과 종료지점을 나타낸다. 프로그램이 실행되는 동안의 전력을 측정하기 위해서는, 이 시작점과 종료지점까지 사용된 전류를 측정해야 한다. ②에서 전압은 3.3V가 일정하게 공급된다. 실제로 그래프 상에서는 차이가 있지만 그 차이가 극소량이기 때문에 3.3V로 측정하였다. ③에서 나오는 데이터 중 Q는은 시작점에서 종료지점까지의 누적 전류량을 나타낸다. 이 누적 전류량을 전력을 구하는 공식인 $P = V \times I$에 대입하면 전력량을 측정할 수 있다.

(그림 2) 전력 측정 결과 그래프

3.1 절차식 언어(C) 패러다임에 대한 저전력 패턴 정의

소프트웨어 코드의 전력을 측정하기 위해 절차식 언어 패러다임에 대한 코드 전력 패턴을 분석하였다. 본 연구에서는 3개의 절차식 코드 패턴을 정의하고 전력을 측정하였다. 장 수 때문에 2개의 패턴은 생략했다. 이 패턴들을 전체 소프트웨어에서 추출하기 위해 Cppcheck와 정규표현식을 사용했다. 또한, 이를 개선하는 패턴을 정의한다.

<표 1> Loop Up과 Loop Down 패턴 측정 결과

Item	Pattern Name	
	Loop Up	Loop Down

Code Pattern	int main(){
	for (int i=0; i<100; i++){
	printf("*");
	}
	return 0;
	}
		int main(){
		for (int i=100; i>0; i--){
		printf("*");
		}
		return 0;
		}

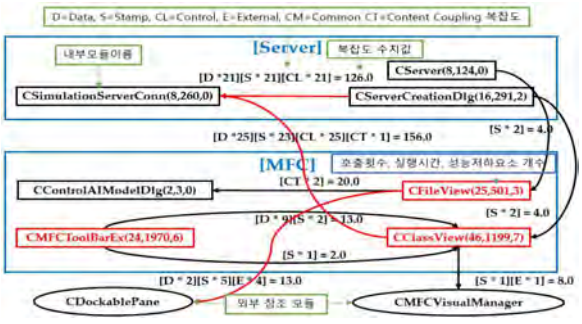
평균 소비전력	5.2670697mW	5.2263594mW
정규 표현식	[a-zA-Z_]([a-zA-Z_0-9])* [&<] ([a-zA-Z_0-9])+ ; [a-zA-Z_]([a-zA-Z_0-9])* \+\+	

표 1은 Loop Up과 Loop Down의 측정 결과이다. Loop Down 패턴의 전력이 더 낮다. 다음은 정규표현식에 대한 설명이다. for문 내부의 조건식에서 'i'는 변수이기 때문에 맨 처음에 숫자가 올 수 없으므로 '[a-zA-Z_]'으로 표현한다. 그리고 그 뒤엔 ([a-zA-Z_0-9])*로 표현하여 변수의 갈아에 상관없이 검출 할 수 있다. '*'는 0개 이상 존재한다는 의미다. 조건식 내부의 값을 비교하기 위해 '[&<]'을 사용했고 조건을 비교하기 위해 변수 혹은 숫자가 올 수 있으므로 이를 '([a-zA-Z_0-9])+'로 표현했다. '+'는 1개 이상 존재한다는 의미다. 증감식 'i++'은 변수를 표현하는 '[a-zA-Z_]([a-zA-Z_0-9])*'뒤에 \+\+를 사용했다.

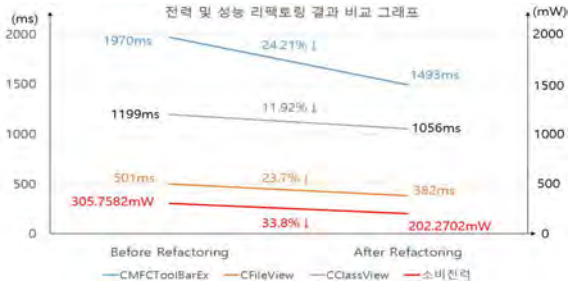
4. 성능개선 코드 가시화

(그림 3)은 다관절 로봇 시뮬레이터 프로그램의 성능측정 결과 그래프이다[4]. 사각형 모듈은 모듈이름(호출횟수(번), 실행시간(ms), 성능저하요소(개))로 이루어져있고 성능저하요소가 3개 이상이면 빨간색으로 표시된다. 동그라미 모듈은 헤더파일로 외부참조 모듈이다. 각 모듈간의 호출 관계는 화살표로 표현하고 호출할 때 발생하는 결합도가 표시된다. 성능가시화 결과 3가지 모듈이 심각한 성능저하요소를 가지고 있다.

그림 4는 리팩토링 전 후를 비교하는 그래프이다. 개선 전 전력량은



(그림 3) 성능 측정 결과 그래프



(그림 4) 전력 및 성능 리팩토링 결과 비교 그래프

305.7582mW이다. 이를 저전력 패턴을 적용한 후의 전력량은 202.2702mW이다. 총 33.8%의 전력감소효과를 얻을 수 있었다. 실행시간도 1493ms, 1056ms, 382ms로 감소했다.

5. 결론 및 향후연구

IoT 기반 서비스 산업의 확장으로 IoT 디바이스들에 대한 수요가 증가하고 있다. 장기간 IoT 디바이스들을 운용하기 위한 저전력 사용 방법 및 성능 개선 등에 대한 연구가 필요하다. 본 연구에서 소프트웨어 코드의 전력 사용 효율화 및 성능 개선 방법을 제안했다. 이 방법은 전체 소스 코드의 복잡도를 가시화 하고 가장 복잡한 모듈들을 개선함으로써, 소프트웨어의 소비전력 감소 및 성능 향상이 가능하다. 향후 연구로는 이번 연구에서 줄이지 못한 모듈간의 호출관계에서 발생하는 결합도를 개선하여 전력을 감소하고 성능을 개선하는 연구를 진행할 것이다.

ACKNOWLEDGE

본 논문은 2019년도 산업통상자원부의 '창의산업융합 특성화 인재양성사업'((과제번호 N0000717)과 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2017R1D1A3B03035421).

참고문헌

[1] Antonio Vetro', Luca Ardito, Giuseppe Procaccianti, Maurizio Morisio, Definition, Implementation and Validation of Energy Code Smells: an Exploratory Study on an Embedded System, The Third International Conference on Smart Grids, 34-39, 2013

[2] 안현식, 이완영, 김영철, 고급 프로그래밍 코드 내 전력 소비 측정 통한 저전력 코드 패턴 메카니즘 식별 가이드, 2019년 ICT플랫폼, 2019, 15-18

[3] 강건희, 박보경, 장우성, 황준순, 권하은, 이한솔, 이현준, 김영철, 소프트웨어 성능 가시화를 위한 툴 체인 개발, 18,1,395-398,2016

[4] Bo Kyung Park, Byungkook jeon, R. Young Chul Kim, Improvement Practices in the Performance of a CPS Multiple-Joint Robotics Simulator, Applied Sciences, 10, 185-198, 2019