


## 2차년도 주요 결과물

(과제명) 대규모 분산 에너지 저장장치 인프라의 안전한  
자율운영 및 성능 평가를 위한 지능형 SW  
프레임워크 개발

(과제번호) 2021-0-00077

- 결과물명 : 시각화 기반의 경향, 특이점, 이상치  
분석 결과서
- 작성일자 : 2022년 12월 7일

과학기술정보통신부 SW컴퓨팅산업원천기술개발사업  
“2차년도 주요 결과물”로 제출합니다.

수행기관	성명/직위	확인
슈어소프트테크(주)	심정민 / 이사	

정보통신기획평가원장 귀하

## 목차

1. 개요
  - 1.1 에너지저장장치의 구성
  - 1.2 BMS의 수행 기능
  - 1.3 BMS 시스템 리소스 수집 목적
2. BMS의 에너지원 및 시스템 리소스 데이터의 수집 및 시각화
  - 2.1 BMS 시스템 리소스 측정 모듈 제작
  - 2.2 BMS 에너지원 및 시스템 리소스 측정/수집
  - 2.3 수집 데이터 전처리 및 시각화
3. 결과

### <표 차례>

표 1 시스템 소스 데이터 측정 결과

### <그림 차례>

- 그림 1 BMS 시스템 리소스 측정 코드  
그림 2 BMS 테스트베드  
그림 3 인셀 테스트베드 환경  
그림 4 Rack 측정 데이터  
그림 5 데이터 처리 전 시각화 그래프  
그림 6 데이터 처리 후 시각화 그래프

## 1. 개요

### 1.1 에너지저장장치의 구성

에너지저장장치(Energy Saving System; ESS)는 전력을 효율적으로 사용하게 하는 설비로, 주파수 조정, 신재생에너지 출력 안정화, 전력피크 저감 등에 활용함으로써 전력품질 향상 및 전력수급 위기에 대응이 가능한 설비이다.

에너지저장장치는 배터리(Battery), 전력변환장치(PCS; Power Conditioner System), 배터리관리시스템(BMS; Battery Management System), 전력관리장치(PMS; Power Management System)으로 구성되어 있다. 전력변환 장치는 입력되는 교류 전기를 직류로 변환하여 배터리 셀에 저장하며 방전 시에는 반대로 직류 전기를 교류로 변환하여 외부에 전력을 공급하는 역할을 한다. 배터리관리시스템은 MCU와 셀 모니터링 칩셋으로 구성 되어 있으며 각 셀에 직접 접근하면서 충/방전 제어 및 상태를 확인하는 역할을 한다. 전력관리장치는 모니터링 및 제어가 가능한 시스템으로 에너지저장장치가 충전과 방전을 수행할 조건을 설정할 수 있게 하여 보다 효율적으로 운영이 가능하도록 제어하는 요소이다.

### 1.2. BMS의 수행 기능

ESS의 구성 요소 중 하나인 BMS는 에너지를 충/방전시키는 배터리 셀을 관리하는 역할을 한다. BMS는 다음의 기능을 수행한다.

#### A. 배터리 셀의 상태 측정 및 저장

BMS 내부의 셀 모니터링 칩에는 GPIO 핀과 ADC가 내장되어 있어 셀의 전압, 전류 및 온도를 측정하고 해당 데이터를 기반으로 충/방전 제어 및 상태 예측을 수행한다.

#### B. 충전상태(SOC; State Of Charge) 및 배터리 잔여수명(State Of Health) 예측

셀에서 측정할 수 있는 데이터는 전압, 전류 및 온도 데이터뿐이다. 따라서 셀 자체에서 정확한 충전율과 잔여 수명 값을 측정 하지 못하지만 측정 전압 및 전류 값을 통하여 셀의 충전 상태와 잔여 수명을 예측할 수 있다. BMS는 셀 모니터링 칩에서 측정한 데이터를 통하여 충전 상태 및 잔여 수명을 예측하는 기능을 포함한다.

#### C. 충/방전 제어

BMS는 셀 모니터링 칩에서 측정한 전압 데이터를 통하여 충/방전 수행 여부를 결정하고 내부 GPIO 핀을 사용하여 셀 충/방전기를 제어함으로써 배터리의 충/방전 제어를 수행한다.

#### D. 셀 밸런싱 제어

BMS는 각 셀의 충전 상태를 확인하여 셀이 고르게 충전될 수 있도록 각 셀을 담당하는 충/방전기를 제어한다.

### 1.3 BMS 시스템 리소스 수집 목적

BMS는 에너지원인 배터리 셀을 직접 제어하므로 제어 시 문제가 생겨 제어 실패가 발생하면 ESS 시스템에 화재가 발생할 수 있다. 실제로도 ESS 화재 사건 발생 시 BMS 동작 오류를 원인으로 제기하는 사례가 여러 번 있었고 22년 10월에 발생한 판교 데이터센터 화재 사고도 UPS 내부의 BMS에서 사고 시점까지도 이상을 감지하지 못했다.

따라서 이번 수행 목적은 시스템 리소스의 이상으로 인하여 BMS의 기능 동작에 문제가 발생하는지의 여부 확인과 시스템 리소스 데이터로 인한 이상 발생 여부를 확인하여 SW 안전 요소 이상 검출 모델의 feature로 활용하려고 한다. 이를 위하여 시스템 리소스 수집 모듈을 개발하여 테스트 베드에 탑재 하였고 배터리 셀과 연계하여 충/방전 시험을 통하여 시스템 리소스의 변동 사항과 비

정상 동작 발생 여부를 확인하였다.

## 2. BMS의 에너지원 및 시스템 리소스 데이터의 수집 및 시각화

### 2.1 BMS 시스템 리소스 측정 모듈 제작

```
1 stack_base = (unsigned)&STACK$Base;           // stack base pointer
2 stack_limit = (unsigned)&STACK$Limit;          // stack limit pointer
3 stack_size = (unsigned)&STACK$Limit - (unsigned)&STACK$Base; // total stack size
4
5 uint32_t max_runtime = 0u;
6 uint32_t max_stackuse = 0u;
7
8
9 uint8_t get_stack_use(void)
10 {
11     uint32_t stack_pt = __get_SP();
12     uint32_t stack_use = (stack_limit - stack_pt);
13     uint8_t stack_percent = (uint32_t)(stack_use / stack_size * 100u);
14
15     if(stack_percent > max_stackuse)
16     {
17         max_stackuse = stack_percent;
18     }
19
20     return (uint8_t)stack_percent;
21 }
22
23 float get_mcu_temp(void)
24 {
25     HAL_ADC_Start(&hadc1);
26     while(HAL_ADC_PollForConversion(&hadc1, 1000) != HAL_OK);
27     uint16_t adc_val = HAL_ADC_GetValue(&hadc1);
28
29     float temperature = adc_val * 3300; // Reading in mV
30     temperature /= 8192; // Reading in Volts
31     temperature /= 1000.0f; // Reading in Volts
32     temperature -= 0.76f; // Subtract the reference voltage at 25°C
33     temperature /= 0.0025; // Divide by slope 2.5mV
34     temperature += 25.0; // Add the 25°C
35
36     return temperature;
37 }
38
39 uint32_t start_measure_runtime(void)
40 {
41     start_timer();
42     uint32_t start_time = get_timer();
43     return start_time;
44 }
45
46 uint32_t stop_measure_runtime(uint32_t start_time)
47 {
48     uint32_t end_time = get_timer();
49     uint32_t runtime = end_time - start_time;
50     stop_timer();
51     if(runtime > max_runtime){
52         max_runtime = runtime;
53     }
54     return runtime;
55 }
56
57 float get_reference_voltage(void)
58 {
59     HAL_ADC_Start(&hadc1);
60     while(HAL_ADC_PollForConversion(&hadc1, 1000) != HAL_OK);
61     uint16_t adc_val = HAL_ADC_GetValue(&hadc1);
62     float vref = (float)(1.21 * 4095) / adc_val;
63     return vref;
64 }
```

<BMS 시스템 리소스 측정 코드>

소프트웨어 코드는 의도치 않은 결함이 발생할 수 있으며 여러 산업분야에서 소프트웨어 결함으로 인한 크고 작은 사고가 여러 번 발생했었다. 따라서 BMS 테스트베드에서 측정할 데이터로 MCU Runtime, Stack Usage, MCU Vref, MCU Temperature 및 System Clock로 정하고 해당 리소스를 측정하기 위한 코드를 다음과 같이 테스트베드 BMS의 소프트웨어에 탑재 하였다.

### 2.2 BMS 에너지원 및 시스템 리소스 측정/수집



<BMS 테스트베드>



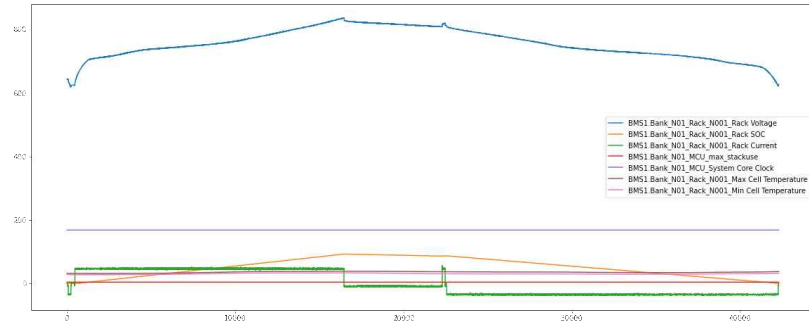
<인셀 테스트베드 환경>

Time	MCU Runtime	Stack Usage	MCU Vref	MCU Temperature	System Clock
2023-09-20 10:00:00	0.00	0.00	1.21	25.00	1000000
2023-09-20 10:00:01	0.00	0.00	1.21	25.00	1000001
2023-09-20 10:00:02	0.00	0.00	1.21	25.00	1000002
2023-09-20 10:00:03	0.00	0.00	1.21	25.00	1000003
2023-09-20 10:00:04	0.00	0.00	1.21	25.00	1000004
2023-09-20 10:00:05	0.00	0.00	1.21	25.00	1000005
2023-09-20 10:00:06	0.00	0.00	1.21	25.00	1000006
2023-09-20 10:00:07	0.00	0.00	1.21	25.00	1000007
2023-09-20 10:00:08	0.00	0.00	1.21	25.00	1000008
2023-09-20 10:00:09	0.00	0.00	1.21	25.00	1000009
2023-09-20 10:00:10	0.00	0.00	1.21	25.00	1000010
2023-09-20 10:00:11	0.00	0.00	1.21	25.00	1000011
2023-09-20 10:00:12	0.00	0.00	1.21	25.00	1000012
2023-09-20 10:00:13	0.00	0.00	1.21	25.00	1000013
2023-09-20 10:00:14	0.00	0.00	1.21	25.00	1000014
2023-09-20 10:00:15	0.00	0.00	1.21	25.00	1000015
2023-09-20 10:00:16	0.00	0.00	1.21	25.00	1000016
2023-09-20 10:00:17	0.00	0.00	1.21	25.00	1000017
2023-09-20 10:00:18	0.00	0.00	1.21	25.00	1000018
2023-09-20 10:00:19	0.00	0.00	1.21	25.00	1000019
2023-09-20 10:00:20	0.00	0.00	1.21	25.00	1000020

<Rack 측정 데이터>

시스템 리소스 측정 코드를 인셀 테스트베드 환경에 구축된 Rack BMS 소프트웨어의 소스 코드에 삽입하고 Rack BMS 충/방전 시험 시 시스템 리소스 데이터를 함께 측정하도록 하였다. 본 테스트베드에서 측정된 데이터는 1차적으로 KETI에서 가공을 수행하여 Google Cloud Storage에 적재된다.

## 2.3 수집 데이터 전처리 및 시각화

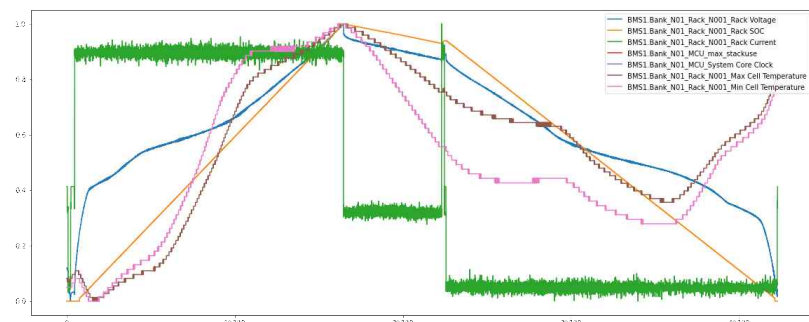


<데이터 처리 전 시각화 그래프>

본 데이터를 도표화시키면 다음과 같다. 이와 같이 순수한 상태의 데이터는 양 데이터 간 값의 범위도 다르고 불필요한 데이터 범례도 있어 이를 해결해야 데이터의 시각화 및 분석이 가능해진다. 해당 전처리 과정은 시각화 및 데이터 연관성 분석을 위하여 불필요한 라벨 제거, 정규화를 수행하였다.

우선 시각화에 긍정적인 영향을 끼치지 않는 라벨을 제거한다. 본 데이터를 측정하면서 발생한 데이터 중 셀 간 에너지원 편차 데이터 발생 지점(Position) 관련 데이터와 당시의 시각을 측정하기 위한 Time 데이터가 있었다. 해당 데이터는 시각화하지 않을 데이터이므로 제거하는 과정을 수행한다.

정규화는 데이터의 범위 차이를 공통의 범위로 변경하는 과정이다. 각각 측정된 데이터는 서로 다른 데이터 범위를 가진다. 이로 인해 데이터 간 시각화 과정에서 측정 범위가 상대적으로 좁은 데이터의 시각화가 힘들어진다. 이를 해결하기 위하여 정규화 과정 중 min-max scaling 기법을 수행하였으며 해당 기법은 모든 데이터를 0 ~ 1의 데이터 영역으로 좁히게 하여 시각화 및 분석을 수행하였을 때 각 데이터 간의 연관성을 쉽게 파악할 수 있게 해준다.



<데이터 처리 후 시각화 그래프>

지금까지의 과정을 거친 후 데이터를 시각화하면 위 그래프가 출력된다.

### 3. 결과

측정 항목	측정 결과	이상 유무
CPU Max Runtime	1331 $\mu$ s (변동 없음)	無
Stack Memory Usage	평균 점유율 48%	無
MCU Reference Voltage	3.003V ~ 3.017V	無
MCU Temperature	57°C ~ 72.67°C	無
System Clock	168MHz	無

표 1 시스템 리소스 데이터 측정 결과

지금까지 8번의 충/방전 시험을 수행하였으며 에너지원 데이터 및 BMS 시스템 리소스 데이터를 측정하였다. 그동안 BMS 시스템 리소스의 오류로 생길 수 있는 문제점을 예방차원에서 체크하였고 이상치를 발견하지는 못했으나 정상적으로 운영하고 있는 것을 면밀히 모니터링하고 검증할 수 있는 기회가 되었다.

아쉽게도, 데이터에서 이상치를 검출할 수 없었다. 정상적인 BMS의 시스템이 운영되고 있어 내부 메모리 및 System Clock, MCU 내부 온도 등의 문제 또한 발생하지 않아 차년도에는 BMS System의 내부 시스템오류를 모사하여 이상치 데이터를 생성, 정상 데이터와 함께 학습 데이터로 사용하여 SW 안전 요소 이상 검출 모델의 feature 요소로 활용 계획에 있다.