ZUML 2.0[1]

(ZU Macro Language)

[1] Язык программирования зарядного устройства

Кратко суть:

По задумке автора, ЗУ работает как программируемый блок питания. Поэтому нужен:

- Интерпретатор ZUML 2.0 для управления железом ЗУ (микропрограмма);
- Макросы для каждого типа аккумуляторов (хранятся в ЕЕПРОМ);
- Оболочка для создания макросов.

Язык программирования ZUML 2.0 – это оригинальный язык для управления зарядом и разрядом аккумуляторов на ЗУ: v2.0, v1.7, v.B6. Интерпретатор языка работает фазами: загружаем фазу и выполняем ее. Каждая фаза это определенное действие: зарядка постоянным током или напряжением, разрядка постоянным током, струйная зарядка или сложный режим с предварительным разрядом перед измерением. В фазе также определены условия окончания фазы. Условия бывают по току, напряжению, температуре и времени.

Программа ZUML 2.0 (далее просто макрос) пишется в текстовом виде на компьютере. Затем макрос переводится в байтовый код, который записывается в постоянную память ЗУ.

Каждый пользователь может самостоятельно обновлять макросы с помощью большого компьютера (без программатора). Макросы хранятся в ЕЕПРОМ они упорядочены. Размер макроса в кодированном виде занимает от 10 до 255 байт. Некоторые команды не обязательно указывать в коде программы, т.к. они принимаются по умолчанию.

Структура макроса:

Структура текстовой записи макроса:

1. Макрос состоит из заголовка, вступления, секции разрядки, секции зарядки и заключения.
2. Заголовок. Отмечается служебным словом «MACRO». Далее в круглых скобках идет название макроса, по возможности, отражающее химический тип. Название макроса используется для именования файла при записи на жесткий диск. На название макроса налагается условие не более 16 символов, не допускаются символы, запрещенные в названиях файлов.
3. Комментарии. В любой строке макроса после символов «//» и до конца строки можно писать все что угодно. Комментарии отсутствуют в байтовом коде макроса.
4. Вступление содержит объявление глобальных констант.
5. Секция разрядки всегда идет после вступления.
6. Секция зарядки идет после секции разрядки.
7. В зависимости от поставленной задачи (только разрядка, тренировка или только зарядка) интерпретатор выбирает какие секции выполнять.
8. Секция разрядки и секция зарядки состоят из циклов и фаз.
9. Фаза — элементарная операция ЗУ по поддержанию постоянного тока или напряжения до выполнения условия остановки.
10. Цикл — операция по повторению одной или нескольких фаз внутри секции.
11. Заключение определяет какие надо подать сигналы при завершении макроса.

Служебные слова	Переменные	Код	Описание
MACRO name		1	Заголовок программы
END_MACRO		2	Конец текста макроса.
DISCHARGE_SECTION		3	Секция разряда. Всегда первая секция. Если отсутствует или не заполнена, то разряд постоянным током 0.1 А.
CHARGE_SECTION		4	Секция заряда, всегда после секции разряда.
END_SECTION		5	Конец секции заряда
CYCLE(n)	3	6	Цикл (3 прохода) Вложения циклов запрещены. Цикл может находиться в секции разрядки и в секции зарядки. Тренировочный цикл работает отдельно на уровне микропрограммы, выполняя секцию разрядки и секцию зарядки по очереди несколько раз.
END_CYCLE		7	Конец цикла, уменьшение переменной цикла на 1 и сравнение с 0. Если 0 завершение цикла.
FAZE		8	Начало фазы.
END_FAZE		9	Конец фазы и переход к следующей фазе.
START_SHABLON=n		20	При запуске макроса вызывать шаблон n.
AKK_TYPE=		21	Тип аккумулятора: 0-Undef, 1-Pb, 2-NiCd, 3-NiMh, 4-LiPo, 5-LiIon, 6-LiS, 7-LiFe, 8-New
SERIES=		22	Количество последовательно соединенных элементов. Влияет на общее напряжение аккумулятора. Выражается в штуках.
CAPACITY=	Input	23	Емкость в миллиамперах в час. Максимальная емкость 65.535 Ач. Заданная емкость должна быть у каждого последовательного элемента. input (емкость вводится в момент запуска). Если емкость введена при

	auto 4200 mAh		запуске и в тексте программы, то auto (емкость определяется автоматически или неважна) 1 < C < 65535 мАч
CONSTANT_I=	 input 2.1C 2.1A	60	Поддерживать ток зарядки на постоянном уровне Ток в амперах введен из шаблона запуска или шаблона работы Ток в емкостях (должна быть введена емкость аккумулятора) Ток в амперах
CONSTANT_V=	 input 2.1S 2.1V	61	Поддерживать напряжение зарядки на постоянном уровне Напряжение введено из шаблона запуска или шаблона работы 2.1в на банку (должно быть введено количество банок) 2.1в всего, независимо от кол-ва банок.
CHARGE_EX(tch, Ich, tdis, Idis, tV)		62	Заряд. Напряжение измеряется после небольшого предварительного разряда. Заряд 30 секунд потом разряд током 0.1C в течение 1 секунды, потом производится измерение напряжения: tV=0 - при отключенной силовой tV=1 - в конце цикла заряда tV=2 - в конце цикла разряда
TRICKLE(I, V1, V2)		63	Дозарядка (струйкой) током 0.1C при 1.8в на банку и выключить при 1.9в на банку.
IF(T > x)STOP_FAZE		81	Если температура аккумулятора больше ...
IF(V < x)STOP_FAZE	 2.1S	82	Если напряжение аккумулятора меньше … Напряжение «на банку»
IF(V > x)STOP_FAZE		83	Если напряжение аккумулятора больше … Напряжение «на банку»
IF(I < x)STOP_FAZE		84	Если ток аккумулятора меньше …
IF(Tc > x)PAUSE_MACRO		85	Если температура радиатора силовой схемы больше ...
IF(Tc < x)CONTINUE_MACRO		86	Если температура радиатора силовой схемы меньше ...
IF(tm > x)STOP_MACRO		87	Если время работы макроса больше 1/10 секунд ...
IF(tf > x)STOP_FAZE		88	Если время работы фазы больше 1/10 секунд ...
IF(DELTA_PEAK > x)STOP_FAZE	 0.5S	89	Если обнаружен дельтапик высотой более … 0.5 в «на банку»
MUSIC(n)		100	Играем мелодию номер n один раз. Можно ставить между фазами.
LIGHT(n)	 100 101 102 103	101	Режим моргания диода — n коротких гудка, пауза. Можно ставить между фазами. Медленно моргает Быстро моргает Горит постоянно Отмаргивает номер фазы

Структура байтовой записи макроса:

1. Макрос хранится в ЕЕПРОМ.
2. Первый байт смещение на следующий макрос.
3. Второй байт смещение на начало программы (сразу после названия макроса и до DISCHARGE_SECTION).
4. Третий байт смещение на секцию зарядки (следующий байт после CHARGE_SECTION)
5. Четвертый байт смещение на завершающие команды (следующий байт после END_SECTION)
6. Пятый байт и еще несколько байт (не более 16), оканчивающихся 0 - это имя макроса, которое отображается в меню выбора.
7. Далее идут команды макроса разрядки, потом команды зарядки.
8. Между фазами и секциями могут встречаться команды. Они выполняются в порядке следования между фазами.
9. При кодировании величины тока значение из макроса умножается на 100 и округляется до целого. Если указано относительное значение (с символом «C»), то к полученному числу добавляется 0x8000. Если указано абсолютное значение с символом «A», то к полученному числу

ничего не добавляется. В код программы сначала вписывается младший байт потом старший.

10. При кодировании величины напряжения значение из макроса умножается на 100 и округляется до целого. Если указано относительное значение (с символом «S»), то к полученному числу добавляется 0x8000. Если указано абсолютное значение с символом «V», то к полученному числу ничего не добавляется. В код программы сначала вписывается младший байт потом старший.
11. Последний байт контрольная сумма.

Стандартные макросы.

Условия по умолчанию:

1. Всегда проверяется перегрев схемы, если это условие не изменено в программе. Если схема перегрелась, все макросы останавливаются и ждут остывания схемы (IF(Tc>70)PAUSE_MACRO=10).
2. Всегда проверяется температура аккумулятора, если это условие не изменено в программе. При перегреве аккумулятора заканчивается текущая фаза (IF(T>60)STOP_FAZE). При превышении скорости роста температуры аккумулятора тоже самое. Датчик температуры учитывается, если он подключен. Если датчик не подключен, то условие по температуре подменяется условием по емкости: если полученная емкость превысила указанную в 1.5 раза, то завершение фазы.
3. Балансир включается автоматически при обнаружении. Про него ни слова в макросе указывать не надо. Пока надобности в ином подходе не возникло.
4. При выборе режима зарядки, фазы разряда игнорируются.
5. При выборе разрядки, фазы зарядки игнорируются.
6. При выборе тренировки, учитывается все.

Заряд Pb:

```
MACRO Pb
      AKK_TYPE=Pb
      DISCHARGE_SECTION
            FAZE
            CONSTANT_I=0.3C
            IF(V<0.9S)STOP_FAZE()
            END_FAZE
      END_SECTION
      CHARGE_SECTION
            FAZE
            CONSTANT_I=INPUT
            IF(V>2.4S)STOP_FAZE()
            END_FAZE
            FAZE
            TRICKLE(0.1C,2.3S,2.5S)
            IF(t>120)STOP_FAZE()
            END_FAZE
      END_SECTION
      MUSIC(STOP)
END_MACRO
```

Свинцовые аккумуляторы обычно массивные. Это означает, что их температуру бесполезно контролировать. В зависимости от требуемой скорости, «свинец» заряжают до определенного напряжения, далее переводят в режим циклической дозарядки.

Для свинца нет тренировки, но для него есть режим десульфатации CHARGE_EX**(60, 0.5C, 3, 0.1C)**, если вы знаете какие параметры надо установить. Я пока не знаю.

Заряд NiCd:

```
MACRO NiCd
      AKK_TYPE=NiCd
      DISCHARGE_SECTION
            FAZE
            CONSTANT_I=1C
            IF(V<0.8S)STOP_FAZE()
            END_FAZE
      END_SECTION
      CHARGE_SECTION
            FAZE
            CONSTANT_I=INPUT
            IF(DELTA_PEAK=0.2S)STOP_FAZE()
            END_FAZE
      END_SECTION
      MUSIC(STOP)
END_MACRO
```

Для случая тренировок предусмотрен цикл по количеству выбранных тренировок. Кадмий подвержен памяти. Перед зарядкой его необходимо разрядить. В фазе заряда заряжаем постоянным током в зависимости от выбранной скорости. Сигналом к окончанию служит дельтапик. По умолчанию проверяется также условие перегрева аккумулятора и скорость роста температуры.

Заряд NiMh

```
MACRO NiMh
      AKK_TYPE=NiMh
      DISCHARGE_SECTION
            FAZE
            CONSTANT_I=1C
            IF(V<1S)STOP_FAZE()
            END_FAZE
      END_SECTION
      CHARGE_SECTION
            FAZE
            CONSTANT_I=INPUT
            IF(DELTA_PEAK=0.2S)STOP_FAZE()
            END_FAZE
      END_SECTION
      MUSIC(STOP)
END_MACRO
```

Почти тоже самое что и для кадмия, кроме минимального напряжения разряда.

Заряд LiPo

```
MACRO LiPo
      AKK_TYPE=LiPo
      DISCHARGE_SECTION
            FAZE
            CONSTANT_I=1C
            IF(V<2.8S)STOP_FAZE()
            END_FAZE
      END_SECTION
      CHARGE_SECTION
            FAZE
            CONSTANT_I=INPUT
            IF(V>4.1S)STOP_FAZE()
            END_FAZE
            FAZE
            CONSTANT_V=4.2S
            IF(I<0.1A)STOP_FAZE()
            END_FAZE
      END_SECTION
      MUSIC(STOP)
END_MACRO
```

Полимер заряжаем фиксированным током до 4.1 вольт на банку, потом поддерживаем фиксированное напряжение пока ток не обнулится.

Заряд LiIon

```
MACRO LiIon
      AKK_TYPE=LiIon
      DISCHARGE_SECTION
            FAZE
            CONSTANT_I=1C
            IF(V<2.8S)STOP_FAZE()
            END_FAZE
      END_SECTION
      CHARGE_SECTION
            FAZE
            CONSTANT_I=INPUT
            IF(V>4S)STOP_FAZE()
            END_FAZE
            FAZE
            CONSTANT_V=4.1S
            IF(I<0.1A)STOP_FAZE()
            END_FAZE
      END_SECTION
      MUSIC(STOP)
END_MACRO
```

Почти тоже самое что и полимер.

Заряд LiFe

```
MACRO LiFe
      AKK_TYPE=LiFe
      DISCHARGE_SECTION
            FAZE
            CONSTANT_I=1C
            IF(V<2.5S)STOP_FAZE()
            END_FAZE
      END_SECTION
      CHARGE_SECTION
            FAZE
            CONSTANT_I=INPUT
            IF(V>3.5S)STOP_FAZE()
            END_FAZE
            FAZE
            CONSTANT_V=3.6S
            IF(I<0.1A)STOP_FAZE()
            END_FAZE
      END_SECTION
      MUSIC(STOP)
END_MACRO
```

Этот вид аккумуляторов ничего не боится, только количество циклов зарядка/разрядка у него ограничено.