Лабораторная работа №7

Умножитель-сумматор. Flash-память

Цель: Изучить принципы работы умножителя-сумматора и flash-памяти, закрепить навыки комплексного использования возможностей микроконтроллера MSP430F5529 и его интегрированных устройств на базе экспериментальной платы MSP-EXP430F5529.

Задача: Написать программу-калькулятор в соответствии с заданием.

7.1 Умножитель-сумматор

Одним из периферийных устройств микроконтроллера MSP430F5529 является умножитель 32 х 32. Аппаратный умножитель поддерживает:

- умножение со знаком и без знака;
- умножение со знаком и без знака с накоплением;
- операнды 8, 16, 24 и 32 бита;
- насыщение;
- дробные числа;
- 8- и 16-битные операции, совместимые с 16-битным аппаратным умножителем.

Размер операндов определяется адресом, в который записан первый операнд, и тем, записан он как слово либо как байт.

Структура умножителя представлена на рисунке 7.1. Аппаратный умножитель имеет два 32-битных регистра операндов – операнд один (OP1) и операнд два (OP2), 64-битный результат доступен через регистры от RES0 до RES3. Для совместимости с 16x16 умножителем, результат 8- или 16-битных операций доступен через регистры RESLO, RESHI, а так же SUMEXT. RESLO содержит младшее слово 16x16-битной операции, RESHI содержит старшее слово результата, и SUMEXT содержит информацию о результате.

Результат 8- и 16-битного умножения вычисляется за три такта MCLK и может быть прочитан следующей инструкцией после записи в OP2, за исключением обращения к результату, используя косвенную адресацию (в этом случае требуется использовать NOP).

Результат 24- и 32-битной операции может быть прочитан после записи OP2 или OP2H начиная с RES0, за исключением косвенной адресации как в 16-битном случае.

Операнд один (OP1) имеет 12 регистров (табл. 7.1), используемых для загрузки данных в умножитель и выбора режима умножения. Запись младшего слова первого операнда в определенный регистр выбирает тип выполняемой операции, но не запускает саму операцию. Когда выполняется запись второго слова в старшую часть регистра с суффиксом 32H, умножитель будет

использовать 32-битный OP1, иначе используется 16-битный операнд. Последний адрес, в который записывается часть первого аргумента, до начала записи второго, определяет размер первого аргумента. Например, если MPY32L записан первым, с последующей записью в MPY32H – все 32 бита используются в качестве OP1. Если MPY32H записан перед записью в MPY32L – умножитель игнорирует MPY32H и использует 16-битный OP1, используя данные из MPY32L.

Повторяющиеся операции умножения могут быть выполнены без перезаписи OP1, если значение OP1 используется для всех них.

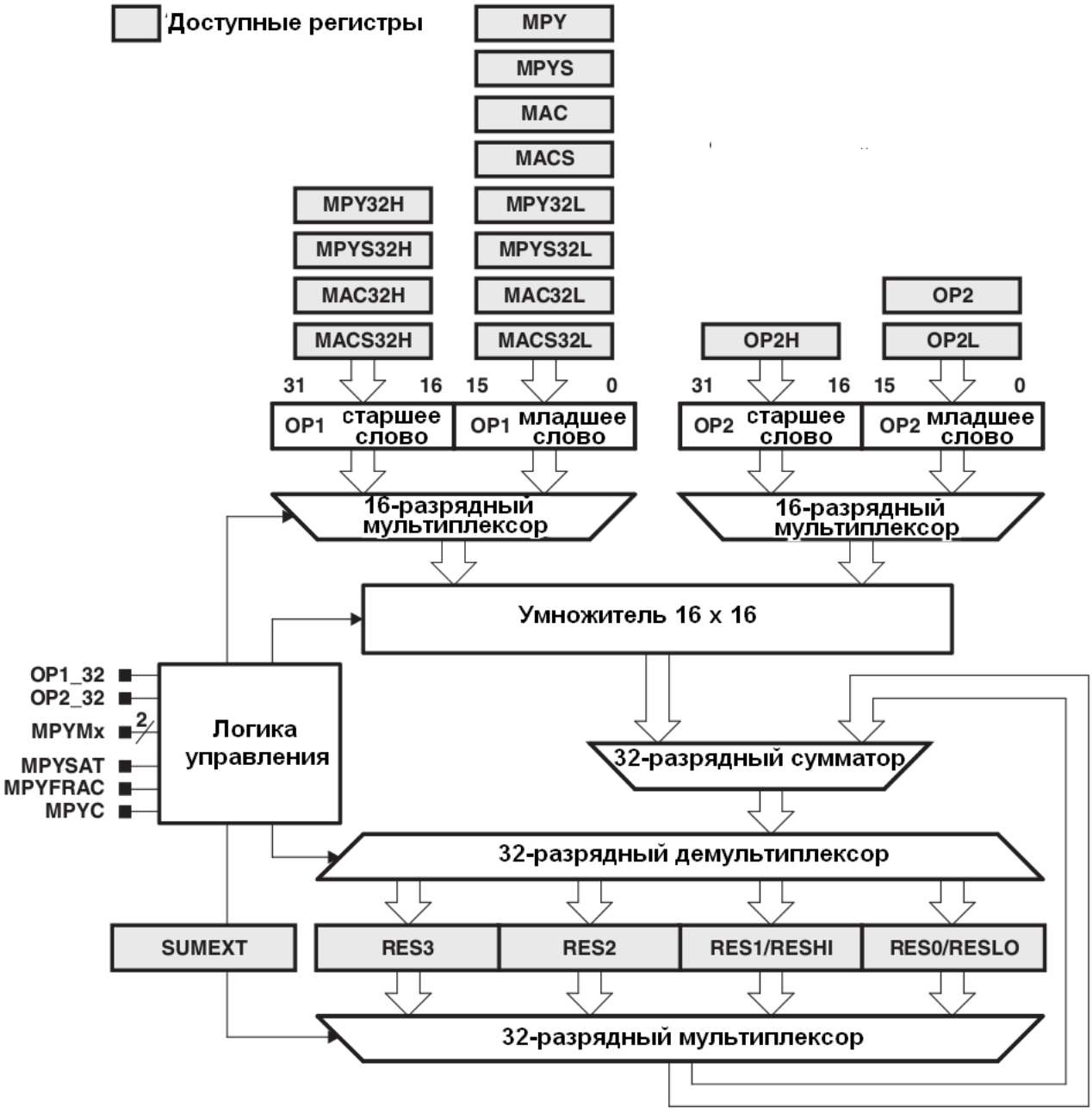

Рис. 7.1 Структура умножителя

Таблица 7.1. Регистры операнда 1

Регистры OP1	**Операция**
MPY	Беззнаковое умножение – биты 0 – 15
MPYS	Знаковое умножение – биты 0 – 15
MAC	Беззнаковое умножение с накоплением – биты 0 – 15
MACS	Знаковое умножение с накоплением – биты 0 – 15
MPY32L	Беззнаковое умножение – биты 0 – 15
MPY32H	Беззнаковое умножение – биты 16 – 31
MPYS32L	Знаковое умножение – биты 0 – 15
MPYS32H	Знаковое умножение – биты 16 – 31
MAC32L	Беззнаковое умножение с накоплением – биты 0 – 15
MAC32H	Беззнаковое умножение с накоплением – биты 16 – 31
MACS32L	Знаковое умножение с накоплением – биты 0 – 15
MACS32H	Знаковое умножение с накоплением – биты 16 – 31

Таблица 7.2. Регистры операнда 2

Регистры OP2	**Операция**
OP2	Запуск выполнения операции с 16-битым OP2 – биты операнда 0 – 15
OP2L	Запуск выполнения операции с 32-битым OP2 – биты операнда 0 – 15
OP2H	Продолжение умножения с 32-битным OP2 – биты операнда 16 – 31

Запись второго операнда в OP2 инициирует операцию умножения. Запись в OP2 запускает выбранную операцию с 16-битным вторым операндом и значением из OP1. Запись в OP2L запускает выполнение выбранной операции с 32-битным вторым операндом и умножитель ожидает записи старшего слова в OP2H. Запись в OP2H без предварительно записанного OP2L игнорируется. Регистры 2 операнда приведены в таблице 7.2.

Результат умножения всегда имеет размер 64 бита. Он доступен в регистрах RES0 – RES3. Для знаковых операций добавляется знаковый бит. Для совместимости с 16x16 умножителями поддерживаются регистры RESLO, RESHI и SUMEXT. Бит MPYC отражает перенос умножителя, как указано в таблице ниже. Содержимое расширенного регистр суммы SUMEXT зависит от операции и приведено в таблице:

Таблица 7.3. Формирование результата

Режим	SUMEXT	MPYC
MPY	Всегда 0000h	Всегда 0
MPYS	Содержит расширенный знак результата: 0000h — результат положительный или 0 0FFFFh — результат отрицательный	Содержит знак результата: 0 — результат положительный или 0; 1 — результат отрицательный
MAC	Содержит перенос: 0000h — нет переноса; 0FFFFh — результат имеет перенос	Содержит перенос: 0 — нет переноса; 1 — результат имеет перенос
MACS	Содержит расширенный знак результата: 0000h — результат положительный или 0; 0FFFFh — результат отрицательный	Содержит перенос: 0 — нет переноса; 1 — результат имеет перенос

Умножитель не может автоматически определить потерю значащих знаков или переполнение в режиме MAC. Например, при 16-битных аргументах и 32-битном результате, диапазон аккумулятора для положительных чисел равен 0 – 7FFF FFFFh, а для отрицательных чисел 0FFF FFFh – 8000 0000h. Переполнение происходит, когда результат суммирования двух отрицательных чисел выходит за диапазон для положительного числа. Потеря значащих разрядов происходит, когда результат сложения двух положительных чисел выходит за диапазон для отрицательного числа. В обоих случаях регистр SUMEXT содержит правильный знак результата: 0FFFFh при переполнении и 0000h при потере значащих разрядов.

7.2 FLASH-память

Карта памяти микроконтроллера приведена в таблице 7.4.

В основной flash-памяти по адресам 00FFFFh–00FF80h (127 b) хранится таблица векторов прерываний. Альтернативная таблица векторов хранится в старших адресах ОЗУ.

Всего на контроллере имеется 128KB flash-памяти. Flash-память может адресоваться и писаться по байтам, по словам и по длинным словам. Однако стирать можно только целый сегмент или все сегменты блока flash-памяти. Основная и информационная память могут использоваться по усмотрению программиста, различия в них заключаются только в размере сегмента. Flash-память имеет 4 сегмента основной памяти (main memory) и 4 сегмента информационной памяти (information memory). Сегменты основной памяти могут быть очищены все вместе или каждый по отдельности, сегменты с информационной памяти могут быть очищены индивидуально, отдельно блокируется запись или стирание для сегмента A. По умолчанию flash-память функционирует в режиме чтения, в котором нельзя писать в память или стирать, память функционирует как ПЗУ. Flash-память является in-system programmable (ISP), т.е. не требует дополнительного внешнего источника

напряжения. Пример организации сегментов flash-памяти приведен на рисунке 7.2.

Таблица 7.4. Карта памяти

Блок памяти	**Сегмент**	**Размер сегмента / диапазон адресов**
Основная flash-память (код)	Bank D	32 KB / 0243FFh–01C400h
	Bank C	32 KB / 01C3FFh–014400h
	Bank B	32 KB / 0143FFh–00C400h
	Bank A	32 KB / 00C3FFh–004400h
ОЗУ (RAM)	Sector 3	2 KB / 0043FFh–003C00h
	Sector 2	2 KB / 003BFFh–003400h
	Sector 1	2 KB / 0033FFh–002C00h
	Sector 0	2 KB / 002BFFh–002400h
USB ОЗУ (RAM)	Sector 7	2 KB / 0023FFh–001C00h
Таблица дескрипторов устройств		147 b / 001A93h–001A00h
Информационная flash-память	Info A	128 B / 0019FFh–001980h
	Info B	128 B / 00197Fh–001900h
	Info C	128 B / 0018FFh–001880h
	Info D	128 B / 00187Fh–001800h
Flash-память начальной загрузки (Bootstrap Loader)	BSL 3	512 B / 0017FFh–001600h
	BSL 2	512 B / 0015FFh–001400h
	BSL 1	512 B / 0013FFh–001200h
	BSL 0	512 B / 0011FFh–001000h
Порты ввода/вывода		4 KB / 000FFFh–0h

В очищенном состоянии значение каждого бита flash-памяти равно 1. Состояние каждого бита может быть изменено с 1 на 0 индивидуально, но обратная операция (из 0 в 1) требует цикла стирания. Наименьшей стираемой единицей данных flash-памяти является один сегмент. Имеется три режима стирания, выбираемых битами ERASE и MERAS.

Конфигурационные регистры FCTLx – это 16-битные защищенные паролем регистры. Считывание или запись состояния регистров осуществляется словами (16 бит). Запись в регистры должна сопровождаться записью пароля 0xA5 в старший байт (в файле msp430f5529.h объявлено макроопределение FWKEY или FWPW). Запись слова в регистр FCTLx со значением в поле пароля, отличным от 0xA5, считается нарушением пароля и приводит к программному перезапуску устройства – сигналу PUC (Power Up Clear). Любое чтение из регистра FCTLx читает 0x96 в старшем байте (макроопределение FRPW или FRKEY). Перед записью или стиранием

необходимо снять бит LOCK. После записи или стирания снять соответствующий бит и установить LOCK. Для выполнения рекомендуется использовать информационную память (Information memory).Для данной работы потребуется использование двух конфигурационных регистров flash-памяти – FCTL1 и FCTL3, поля которых приведены в таблицах 7.5 и 7.6.

Заголовочный файл HAL_FLASH.h определяет следующие функции для работы с flash-памятью:

*void Flash_SegmentErase(uint8_t *Flash_ptr)* — стирает один сегмент flash-памяти, передаваемый указатель должен быть из этого сегмента.

*uint8_t Flash_EraseCheck(uint8_t *Flash_ptr, uint16_t len)* — проверка очистки сегмента. Первый параметр — указатель внутри сегмента, второй параметр — длина проверяемой области.

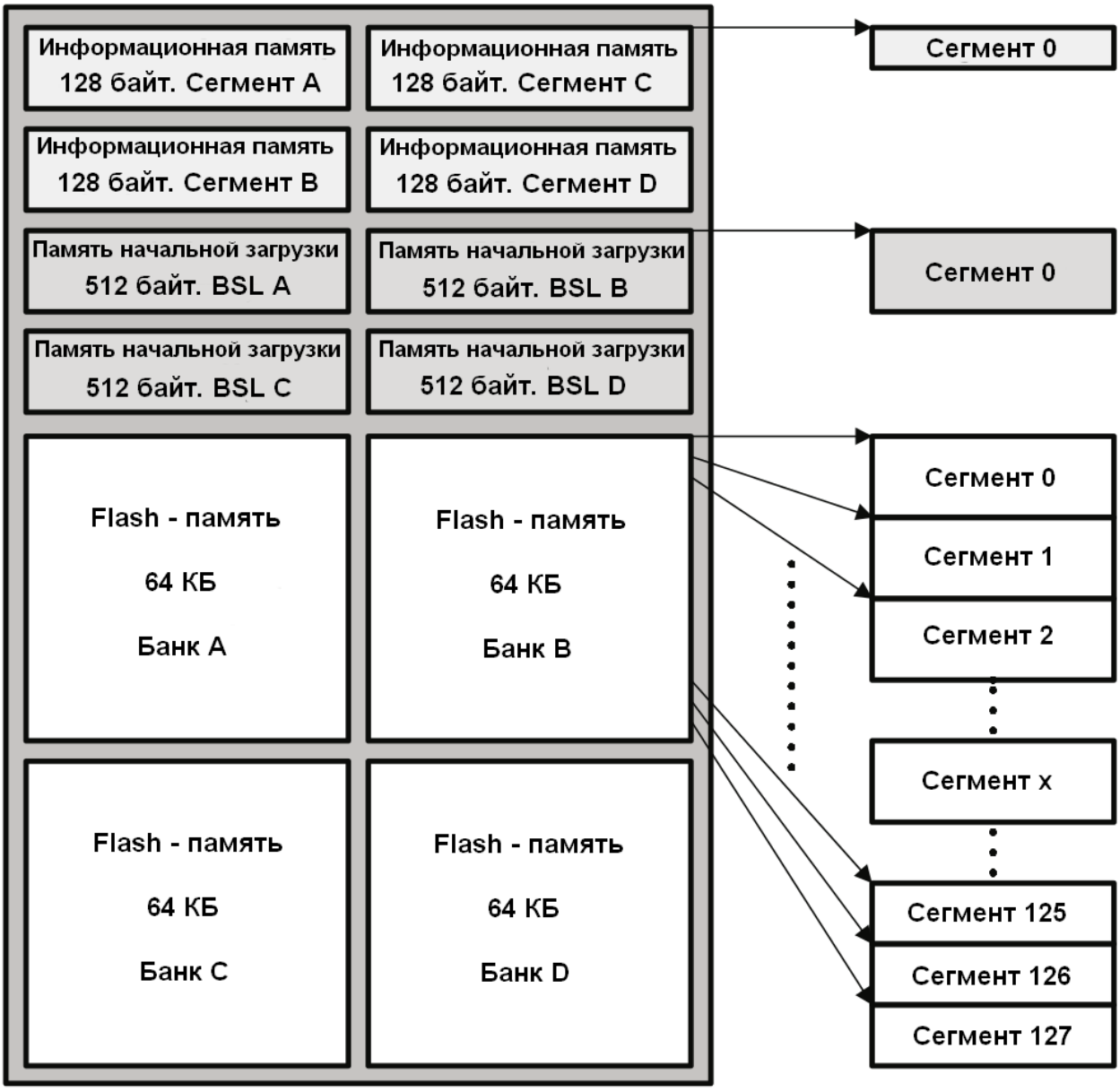

Рис. 7.2 Пример организации сегментов 256КБ flash-памяти

*void FlashWrite_8(uint8_t *Data_ptr, uint8_t *Flash_ptr, uint16_t count);*
*void FlashWrite_16(uint16_t *Data_ptr, uint16_t *Flash_ptr, uint16_t count);*
*void FlashWrite_32(uint32_t *Data_ptr, uint32_t *Flash_ptr, uint16_t count)* — три данных функции записывают соответственно count байт, слов либо двойных слов во flash-память. Первый параметр — указатель на передаваемые данные, второй — указатеьл на область внутри flash-памяти для записи данных, третий — количество передаваемых данных.

*void FlashMemoryFill_32(uint32_t value, uint32_t *Flash_ptr, uint16_t count)* — заливка памяти одинаковым значением (в формате двойного слова), первый параметр — устанавливаемое значение, второй параметр — указатель на область памяти для записи, третий — количество записываемых данных.

Таблица 7.5. Поля регистра FCTL3

Бит	Поле	Тип	Сброс	Описание
15-8	FRPW/FWPW	RW	96h	FCTL пароль. Читается всегда 0x96. Записываться должно 0xA5, иначе генерируется сигнал PUC.
6	LOCKA	RW	1h	Блокировка сегмента A. Запись 1 в этот бит меняет его состояние. Запись 0 не влияет. 0 = Сегмент A информационной памяти разблокируется и может быть записан или стерт в режиме стирания сегмента. 1 = Сегмент A информационной памяти заблокирован и не может быть записан или стерт
5	EMEX	RW	0h	Аварийный выход. Установка бита останавливает операцию стирания или записи. Бит LOCK установлен. 0 = разблокировано, 1 = заблокировано
4	LOCK	RW	1h	Блокировка. Этот бит разблокирует flash-память для записи или стирания. Бит LOCK может быть установлен в момент записи или стирания байта или слова и операция завершится нормально. В режиме записи блока, если LOCK бит установлен пока BLKWRT = WAIT = 1, BLKWRT и WAIT сбрасываются и режим завершается нормально. 0 = разблокировано, 1 = заблокировано
3	WAIT	R	1h	Ожидание. Флаг показывает, что flash-память находится в процессе записи. 0 = flash-память не готова для следующего байта или слова, 1 = flash-память готова для следующего байта или слова
2	ACCVIFG	RW	0h	Флаг прерывания нарушения доступа: 0 = нет прерывания, 1 = прерывание
1	KEYV	RW	0h	Нарушение пароля. Бит сигнализирует, что FCTLx пароль, записанный в какой-либо регистр, сгенерировал PUC. KEYV должен быть сброшен программно. 0 = пароль FCTLx был введен корректно, 1 = пароль FCTLx был введен некорректно
0	BUSY	RW	0h	Занято. Этот бит сигнализирует, что flash-память занята процессом стирания или записи. 0 = не занято, 1 = занято

Таблица 7.6. Поля регистра FCTL1

Бит	Поле	Тип	Сброс	Описание
15-8	FRPW/FWPW	RW	96h	FCTL пароль. Читается всегда 0x96. Записываться должно 0xA5, иначе генерируется сигнал PUC.
7	BLKWRT	RW	0h	Запись блока. BLKWRT и WRT используются совместно, чтобы выбрать режим записи. Значения указаны для пары BLKWRT-WRT: 00 = зарезервировано, 01 = запись байта или слова, 10 = запись длинного слова, 11 = запись блока длинными словами
6	WRT	RW	0h	
5	SWRT	RW	0h	Режим Smart write
2	MERAS			Полное стирание. MERAS и ERASE используются совместно для выбора режима стирания. MERAS и ERASE автоматически сбрасываются когда EMEX в FCTL3 установлен или операция стирания завершена. Значения указаны для пары MERAS-ERASE: 00 = нет стирания, 01 = стирание сегмента, 10 = стирание банка, 11 = полное стирание (очистка всех банков flash-памяти)
1	ERASE			

Задание

В соответствии с вариантом написать программу-калькулятор, поддерживающий 4 арифметических действия с целыми числами, в том числе с отрицательными. Разрядность чисел и система счисления — в соответствии с заданием. Кнопки калькулятора и числа выводятся на ЖКИ экран. Способ выбора кнопки калькулятора на экране определяется в соответствии с заданием, нажатие кнопки калькулятора фиксируется кнопкой S1. Операция умножения должна быть реализована только с использованием встроенного умножителя. В калькуляторе предусмотреть кнопки MS и MR, которые сохраняют (либо восстанавливают) значение числа во flash-память. Допускается использовать любые заголовочные файлы помимо msp430.h, а также использовать высокоуровневые библиотеки (упрощенный вариант задания). Усложненный вариант задания не позволяет пользоваться иными заголовочными файлами помимо msp430.h, а также использовать высокоуровневые библиотеки.

Таблица 7.4. Варианты задания

№ вари-анта	Система счисления	Разрядность чисел	Сегмент flash памяти	Способ выбора кнопки калькулятора на экране
1	Двоичная	2	info_A	Акселерометр
2	Десятичная	2	info_B	Сенсорная клавиатура + компаратор
3	Шестнадцатиричная	2	info_C	Потенциометр + компаратор
4	Двоичная	3	info_D	Потенциометр + АЦП
5	Десятичная	3	info_A	Сенсорная клавиатура + АЦП

№ вари-анта	Система счисления	Разрядность чисел	Сегмент flash памяти	Способ выбора кнопки калькулятора на экране
6	Шестнадцатиричная	3	info_B	Акселерометр
7	Двоичная	4	info_C	Сенсорная клавиатура + компаратор
8	Десятичная	4	info_D	Потенциометр + компаратор
9	Шестнадцатиричная	4	info_A	Потенциометр + АЦП
10	Двоичная	4	info_B	Сенсорная клавиатура + АЦП
11	Десятичная	4	info_C	Акселерометр
12	Шестнадцатиричная	2	info_D	Сенсорная клавиатура + компаратор
13	Двоичная	3	info_A	Потенциометр + компаратор
14	Десятичная	2	info_B	Потенциометр + АЦП
15	Шестнадцатиричная	3	info_C	Сенсорная клавиатура + АЦП