

Национальный технический университет
«Харьковский политехнический институт»

Кафедра «Промышленная и биомедицинская электроника»

*Практическое занятие по дисциплине
«Микропроцессорная техника»*

*Знакомство со средой разработки Keil.
Группа команд передачи данных*

Б.А. Стысло

г. Харьков, 2014 г.

ЗНАКОМСТВО СО СРЕДОЙ РАЗРАБОТКИ KEIL-51.



KEIL™
Tools by ARM

μVision®4
Integrated Development Environment

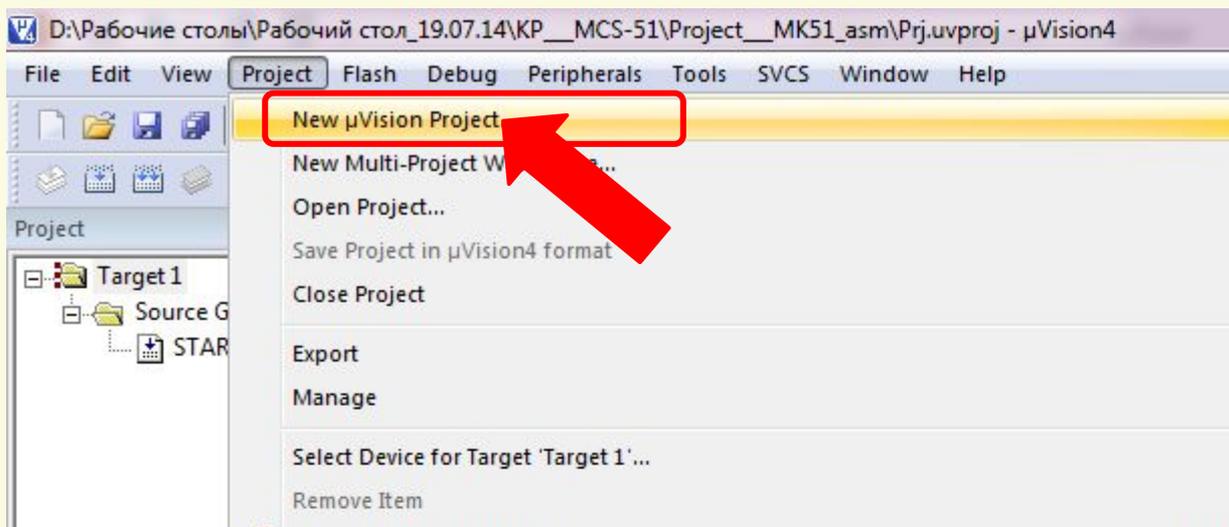
KEIL is a trademark and uVision is a registered trademark of ARM Ltd. All rights reserved.
This product is protected by US and international laws.

The image features a blue and white logo for KEIL Tools by ARM and μVision®4 Integrated Development Environment. The background includes a stylized globe and various colored squares and circles connected by lines, suggesting a network or data flow.

ЗНАКОМСТВО СО СРЕДОЙ РАЗРАБОТКИ KEIL-51.

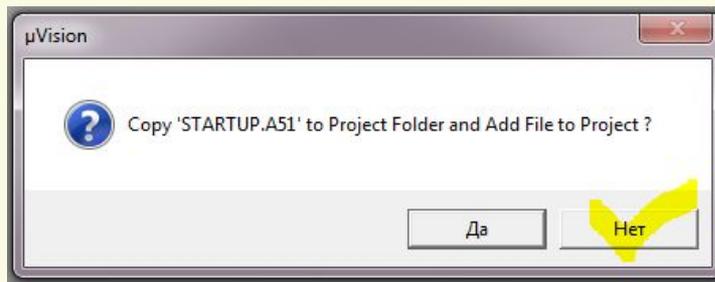
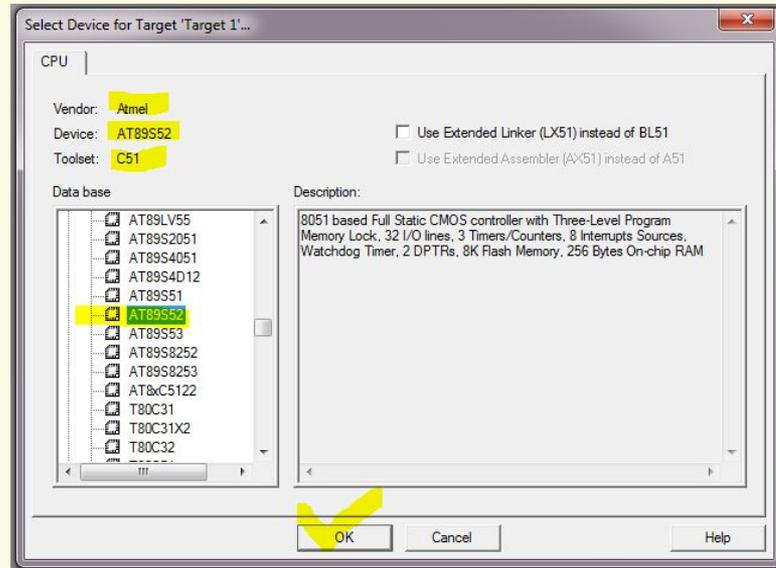
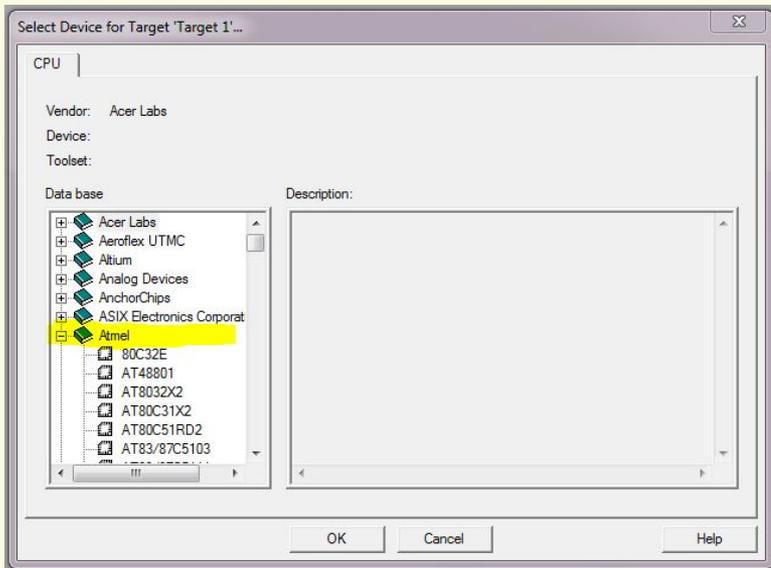
Создание *проекта*

В отличие от используемых ранее средств разработки (Borland Pascal, Borland C), где весь текст программы хранился в одном файле (*.pas), Keil предполагает создание *проекта*, который может содержать множество файлов, в т.ч. библиотеки.



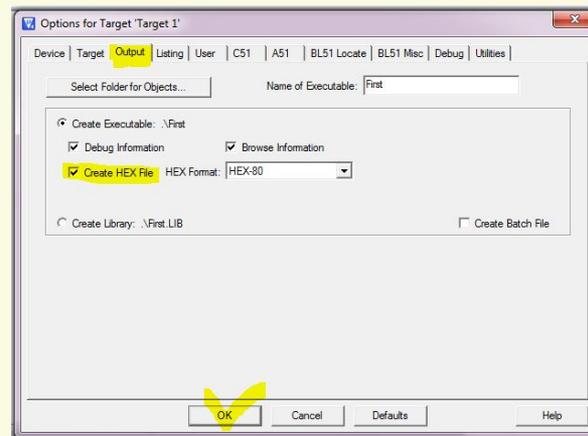
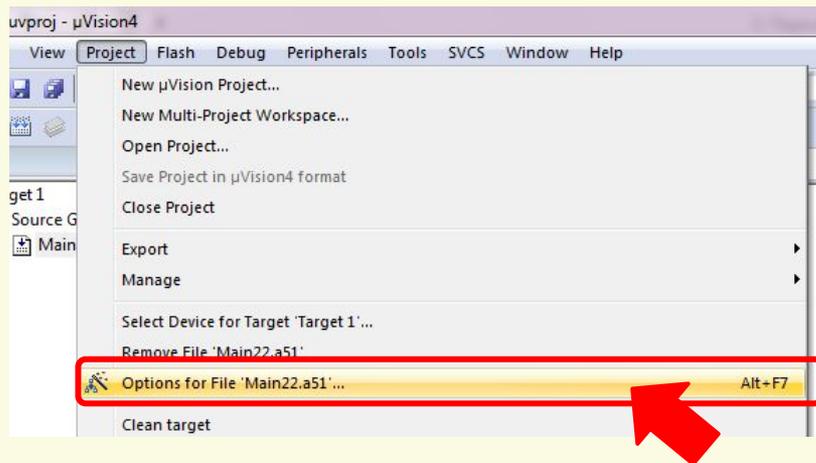
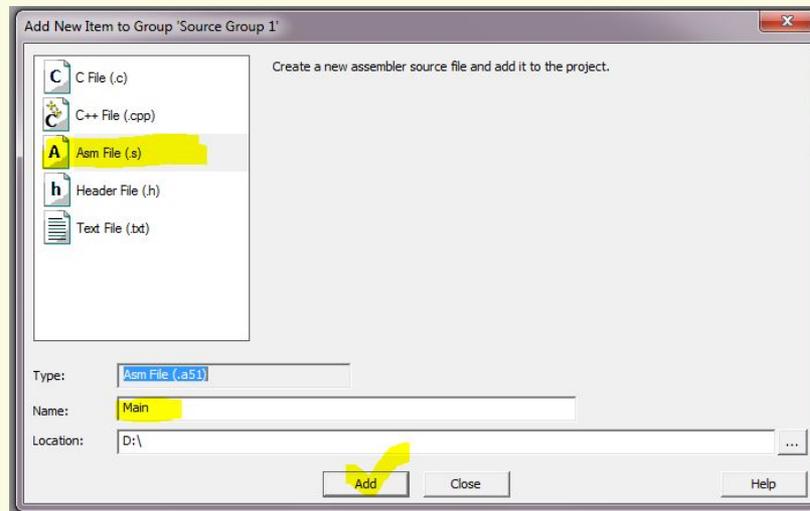
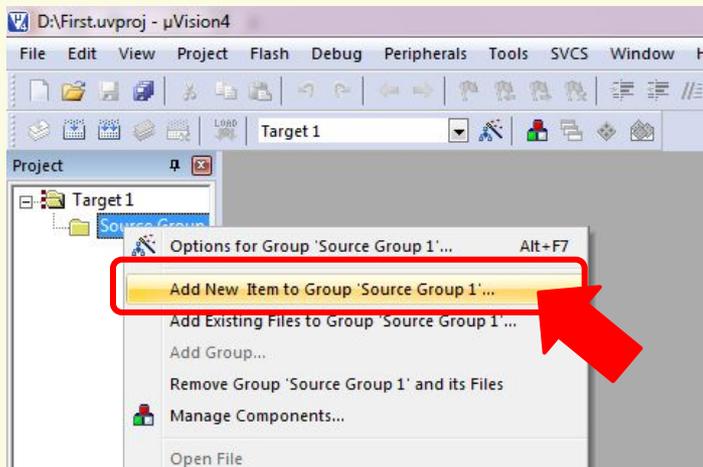
ЗНАКОМСТВО СО СРЕДОЙ РАЗРАБОТКИ KEIL-51.

Создание проекта



ЗНАКОМСТВО СО СРЕДОЙ РАЗРАБОТКИ KEIL-51.

Создание проекта



ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

Данные в МК могут храниться:

- *Регистры (A, B, R0..R7);*
- *Внутренняя память данных (ОЗУ);*
- *Внутренняя память программы (ПЗУ);*
- *Внешняя память данных (ОЗУ);*
- *Внешняя память программы (ПЗУ);*

Регистры – ячейки памяти внутри МК, обмен информации между которыми осуществляется простыми и короткими командами.

Аналогия с языком высокого уровня Pascal:

Регистр = переменная типа byte

R0..R7 – регистры общего назначения (РОН)

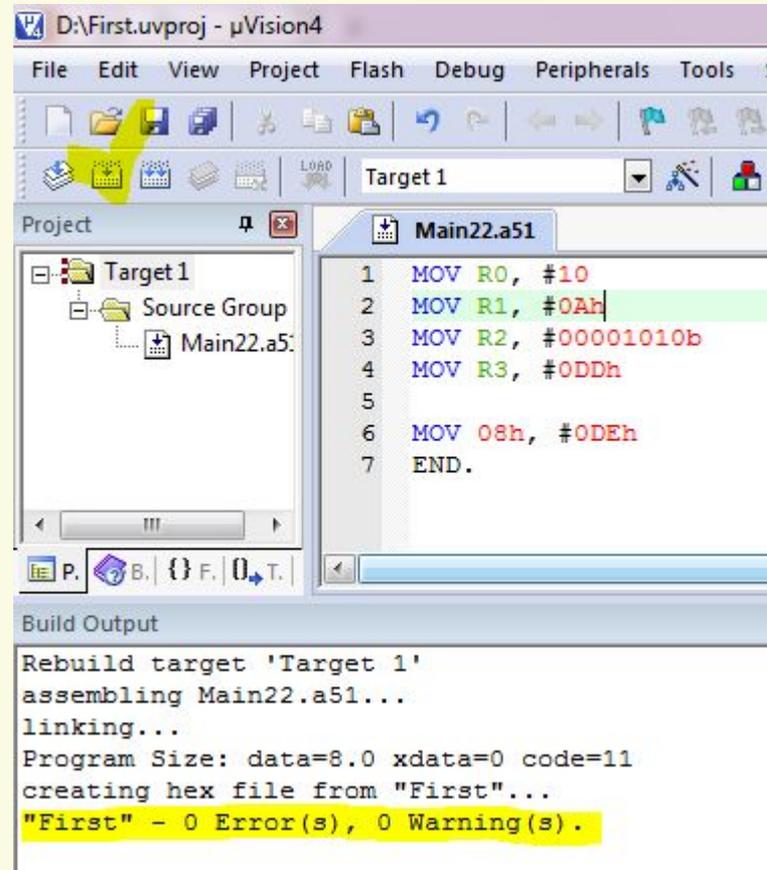
Часть команд возможна лишь с использованием специального регистра – аккумулятора (A)

“HELLO, WORLD!”

Первая программа

```
MOV R0, #10
MOV R1, #0Ah
MOV R2, #00001010b
MOV R3, #0DDh
```

```
MOV 08h, #0DEh
END.
```

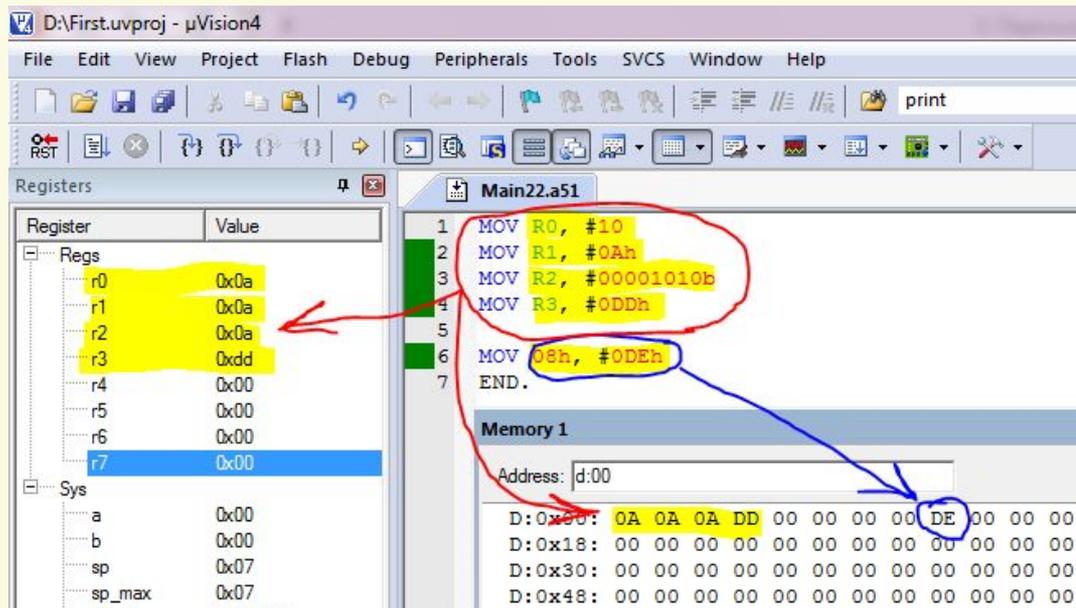


ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

Название команды	Мнемокод	КОП	Г	Б	Ц	Операция
Пересылка в аккумулятор из регистра ($n=0\div7$)	MOV A, Rn	11101rr	1	1	1	$(A) \leftarrow (Rn)$
Пересылка в аккумулятор прямоадресуемого байта	MOV A, ad	11100101	3	2	1	$(A) \leftarrow (ad)$
Пересылка в аккумулятор байта из РПД ($i=0,1$)	MOV A, @Ri	1110011i	1	1	1	$(A) \leftarrow ((Ri))$
Загрузка в аккумулятор константы	MOV A, #d	01110100	2	2	1	$(A) \leftarrow \#d$
Пересылка в регистр из аккумулятора	MOV Rn, A	11111rr	1	1	1	$(Rn) \leftarrow (A)$
Пересылка в регистр прямоадресуемого байта	MOV Rn, ad	10101rr	3	2	2	$(Rn) \leftarrow (ad)$
Загрузка в регистр константы	MOV Rn, #d	01111rr	2	2	1	$(Rn) \leftarrow \#d$
Пересылка по прямому адресу аккумулятора	MOV ad, A	11110101	3	2	1	$(ad) \leftarrow (A)$
Пересылка по прямому адресу регистра	MOV ad, Rn	10001rr	3	2	2	$(ad) \leftarrow (Rn)$
Пересылка прямоадресуемого байта по прямому адресу	MOV add, ads	10000101	9	3	2	$(add) \leftarrow (ads)$
Пересылка байта из РПД по прямому адресу	MOV ad, @Ri	1000011i	3	2	2	$(ad) \leftarrow ((Ri))$
Пересылка по прямому адресу константы	MOV ad, #d	01110101	7	3	2	$(ad) \leftarrow \#d$
Пересылка в РПД из аккумулятора	MOV @Ri, A	1111011i	1	1	1	$((Ri)) \leftarrow (A)$
Пересылка в РПД прямоадресуемого байта	MOV @Ri, ad	0110011i	3	2	2	$((Ri)) \leftarrow (ad)$
Пересылка в РПД константы	MOV @Ri, #d	0111011i	2	2	1	$((Ri)) \leftarrow \#d$
Загрузка указателя данных	MOV DPTR, #d16	10010000	13	3	2	$(DPTR) \leftarrow \#d16$
Пересылка в аккумулятор байта из ПП	MOVC A, @A+DPTR	10010011	1	1	2	$(A) \leftarrow ((A) + (DPTR))$
Пересылка в аккумулятор байта из ПП	MOVC A, @A+PC	10000011	1	1	2	$(PC) \leftarrow (PC)+1, (A) \leftarrow ((A)+(PC))$
Пересылка в аккумулятор байта из ВПД	MOVX A, @Ri	1110001i	1	1	2	$(A) \leftarrow ((Ri))$
Пересылка в аккумулятор байта из расширенной ВПД	MOVX A, @DPTR	11100000	1	1	2	$(A) \leftarrow ((DPTR))$
Пересылка в ВПД из аккумулятора	MOVX @Ri, A	1111001i	1	1	2	$((Ri)) \leftarrow (A)$
Пересылка в расширенную ВПД из аккумулятора	MOVX @DPTR, A	11110000	1	1	2	$((DPTR)) \leftarrow (A)$
Загрузка в стек	PUSH ad	11000000	3	2	2	$(SP) \leftarrow (SP) + 1, ((SP)) \leftarrow (ad)$
Извлечение из стека	POP ad	11010000	3	2	2	$(ad) \leftarrow (SP), (SP) \leftarrow (SP) - 1$
Обмен аккумулятора с регистром	XCH A, Rn	11001rr	1	1	1	$(A) \leftrightarrow (Rn)$
Обмен аккумулятора с прямоадресуемым байтом	XCH A, ad	11000101	3	2	1	$(A) \leftrightarrow (ad)$
Обмен аккумулятора с байтом из РПД	XCH A, @Ri	1100011i	1	1	1	$(A) \leftrightarrow ((Ri))$
Обмен младших тетрад аккумулятора и байта РПД	XCHD A, @Ri	1101011i	1	1	1	$(A_{0..3}) \leftrightarrow ((Ri)_{0..3})$

“HELLO, WORLD!”

Первая программа



MOV R0, #10
MOV R1, #0Ah
MOV R2, #00001010b
MOV R3, #0DDh

MOV 08h, #0DEh
END.

Команда **MOV** выполняет пересылку данных из второго операнда в первый.

ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

MOV R0, #10

MOV R1, #20

MOV A, R0 ; поместить в аккумулятор содержимое R0

MOV 08h, #0DEh

MOV R5, 08h ; поместить в R5 содержимое ячейки ОЗУ с адресом 08h

END.

ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

Косвенная адресация

Косвенный способ адресации предполагает указание операндов посредством адреса, содержащегося в регистре либо в регистровой паре. В команде указывается регистр, который в свою очередь указывает адрес операнда.

```
MOV 08h, #0DEh
```

```
MOV R0, #08h
```

```
MOV A, @R0 ; поместить в аккумулятор содержимое ячейки ОЗУ , адрес  
которой хранится в регистре R0
```

```
END.
```

ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

Карта памяти AT89S52

Адрес байта	Адреса битов								Адрес байта	Адреса битов															
27	3F	3E	3D	3C	3B	3A	39	38	7F	Область памяти общего назначения															
26	37	36	35	34	33	32	31	30																	
25	2F	2E	2D	2C	2B	2A	29	28																	
24	27	26	25	24	23	22	21	20																	
23	1F	1E	1D	1C	1B	1A	19	18																	
22	17	16	15	14	13	12	11	10																	
21	0F	0E	0D	0C	0B	0A	09	08																	
20	07	06	05	04	03	02	01	00																	
1F	Банк 3								2F									7F	7E	7D	7C	7B	7A	79	78
18									Банк 2									2E	77	76	75	74	73	72	71
17																	2D	6F	6E	6D	6C	6B	6A	69	68
10	Банк 1								2C	67	66	65	64	63	62	61	60								
0F									2B	5F	5E	5D	5C	5B	5A	59	58								
08	Банк 0 (по умолчанию) Регистры R0-R7								2A	57	56	55	54	53	52	51	50								
07									29	4F	4E	4D	4C	4B	4A	49	48								
00									28	47	46	45	44	43	42	41	40								

Область памяти с битовой адресацией

ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

Карта памяти AT89S52 (продолжение)

Адрес байта	Адреса битов								Адрес байта	Адреса битов									
98	9F	9E	9D	9C	9B	9A	99	98	SCON	FF									
										F0	F7	F6	F5	F4	F3	F2	F1	F0	B
90	97	96	95	94	93	92	91	90	P1										
										E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
8D	Не адресуется побитово								TH1										
8C	Не адресуется побитово								TH0	D0	D7	D6	D5	D4	D3	D2	-	D0	PSW
8B	Не адресуется побитово								TL1										
8A	Не адресуется побитово								TL0	B8	-	-	-	BC	BB	BA	B9	B8	IP
89	Не адресуется побитово								TMOD										
88	8F	8E	8D	8C	8B	8A	89	88	TCON	B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
87	Не адресуется побитово								PCON										
										A8	AF	-	-	AC	AB	AA	A9	A8	IE
83	Не адресуется побитово								DPH										
82	Не адресуется побитово								DPL	A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
81	Не адресуется побитово								SP										
80	87	86	85	84	83	82	81	80	P0	99	Не адресуется побитово								SBUF

ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

Регистр слова состояния PSW

PSW



RS1:RS0 = 00
RS1:RS0 = 01
RS1:RS0 = 10
RS1:RS0 = 11

Выбор банка регистров

RS1	RS0	Границы адресов R0-R7
0	0	00h-07h
0	1	08h-0Fh
1	0	10h-17h
1	1	18h-1Fh

ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

Чтение и вывод информации в порт

MOV P1, #0FFh ; настроили порт P1 на ввод
MOV P2, #00h ; настроили порт P2 на вывод
MOV P2, P1 ; копируем содержимое P1 в P2
END.

Обращение к регистрам по адресу:

MOV 90h, #0FFh ; настроили порт P1 на ввод
MOV 0A0h, #00h ; настроили порт P2 на вывод
MOV 0A0h, 90h ; копируем содержимое P1 в P2
END.

ГРУППА КОМАНД ПЕРЕДАЧИ ДАННЫХ

Задание для самостоятельного выполнения

Реализовать программу, в результате которой содержимое регистров **R0-R7*** копируется в область ОЗУ по адресу **18h..1Fh**, а затем обнуляется.

**Изначальное содержимое регистров – произвольное (отличное от 00h)*



The End.

**Thank you for
your attention!**