

Harbin Institute of Technology

《微机与微控制器原理》课

实验报告

单片机软件设计实验

姓 名： 哈哈哈

学 号： 嘿嘿嘿嘿嘿嘿嘿嘿

组 号：

班 级： 嘻嘻嘻嘻嘻嘻嘻

指导教师：

哈 尔 滨 工 业 大 学

实践环节评分及标准

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 考察项目 | 考察内容 | | 教师  评价 | 备注 |
| A预习  100分 | 预习认真，掌握实验目的、实验要求和实验内容，设计思路清晰、流程图完整正确（90-100分） | |  | 预习报告：  手写实验预习报告,拍照形成word提交。 |
| 预习较认真，较了解实验目的、实验要求和实验内容，设计思路较清晰、流程图较完整（80-89分） | |
| 预习深度不够，清楚实验目的、实验要求或实验内容，有一定设计思路或部分流程图（60-79分） | |
| 未充分预习，无设计思路，无流程图（60分以下） | |
| B过程表现100分 | 态度认真、操作规范且正确、实验过程完整，进行单步调试或断点调试程序（90-100分） | |  | 每个实验录制一个实验项目讲解视频：含实验内容、操作过程、调试运行结果及分析等。 |
| 态度认真、操作正确、实验过程完整，进行程序调试（80-89分） | |
| 态度认真、操作基本正确、实验过程较完整（60-79分） | |
| 态度不端正、操作不规范、实验过程不完整（60分以下） | |
| C结果验收100分 | 实验结果正确、结论准确，结果分析合理详尽（90-100分） | |  |
| 实验结果正确、结论准确，结果分析合理（80-89分） | |
| 实验结果正确、结论基本正确，无结果分析（60-79分） | |
| 实验结果有错误，未达到实验目的，无结果分析（60分以下） | |
| D程序代码100分 | 程序完整、代码正确、代码规范、有注释（90-100分） | |  | 附实验报告的附录1中,  工程文件按组提交一份 |
| 程序完整、代码正确、代码较规范、有注释（80-89分） | |
| 程序完整，但代码有少量错误，代码不规范（60-79分） | |
| 程序不完整，代码有错误，代码不规范（60分以下） | |
| E实验报告内容100分 | 按照模板实验报告格式规范、内容完整、过程详尽、结果正确、结论准确，结果分析正确（90-100分） | |  | 按模板撰写，每人提交 |
| 按照模板实验报告格式较规范、内容完整、过程清晰、结果正确、结论准确，结果分析分析合理（80-89分） | |
| 按照模板实验报告格式较规范、内容完整、过程清晰、结果正确、结论准确，分析基本正确（60-79分） | |
| 格式不规范、内容不完整（60分以下） | |
| 综合评分 |  | | | |
| 教师评语 |  | 指导教师签字 | |  |

**实验一 单片机软件设计实验任务要求**

|  |  |  |  |
| --- | --- | --- | --- |
| 实验名称 | 单片机软件设计实验 | 实验学时 | 3学时 |
| 实验目的 | 掌握单片机最小系统设计，并对单片机最小系统进行汇编语言程序设计。利用Proteus软件调试仿真完成，通过实验项目设计与程序调试，使学生熟悉8051单片机的指令系统，了解软件设计过程，掌握汇编语言设计方法以及如何使用实验系统提供的调试手段来排除软件错误，锻炼解决软件设计与调试的能力。 | | |
| 实验要求 | 1. 分组进行: 2-3人一组。 2. 实验内容参考实验指导书，实验课前需了解实验目的、要求和实验内容，提前进行程序设计。熟悉PROTUES软件使用，完成实验操作，完成全部实验过程。 3. 提交预习报告：实验课前需完成并提交预习报告，作为预习考核成绩，每人提交，命名：学号姓名-实验名称-预习报告。 4. 提交实验项目讲解视频：由组员共同完成讲解，作为过程及验收考核成绩。按组提交一份。 5. 提交实验代码及工程文件：调试通过后可运行版，作为验收考核成绩，按组提交一份。 6. 提交实验报告：内容及形式按照统一的报告格式（见实验报告模板），作为报告考核成绩，每人提交，命名：学号姓名-实验名称-实验报告。 7. 每个实验报告中的思考题必须完成；对实验中发现的问题应加以讨论，并提出自己的改进意见和要求、希望等。 | | |
| 实验内容 | 项目1数据块传送实验：对单片机外部RAM进行数据块传送操作。  项目2二进制转ASCII码实验：对单片机内部RAM中存储的二进制数进行ASCII码转换操作。 | | |

## 项目1 数据块传送实验

一、实验目的

掌握单片机最小系统的组成。

掌握外部扩展RAM中的数据操作。

二、实验内容

使用AT89C52单片机、锁存器（74LS373）和外部RAM存储器（62256），对单片机外部RAM进行数据块传送操作，实验内容如下：

1. 对2040H地址开始的100个字节进行赋值，值为#5AH；

2. 再将2040H地址开始的20个字节传送到以0030H开始的目的地址。

三、实验原理及程序流程设计

**3.1 实验原理图功能及模块描述**

**1.实验原理详细说明**

将AT89C52单片机、锁存器（74LS373）和外部RAM存储器（62256）进行连接，并通过代码实现对外部RAM进行数据块传送操作。

**2 实验功能模块描述**

如图1所示，从左至右依次为置位电路、时钟振荡电路、8051单片机、锁存器和64位外部RAM。

图表, 示意图, 条形图

描述已自动生成

图1 实验原理图

**3.2 实验程序设计及流程图**

1. 实验程序设计思路：

首先，预设数据，确定循环的起始地址及长度，并进行赋值。

接着，先对高八位IO口（P2）清零，确定传输数据的地址，将数据传至目的地址。

2. 实验流程图如图2所示。



图2 程序流程图

四、实验步骤及结果分析

**4.1 实验步骤说明**

步骤1：对1060H地址开始的100个字节进行赋值，值为#5AH

代码如下

Start:

; Write your code here

MOV A,#5AH ;确定要附的值

MOV DPTR, #1060H ;起始地址送至DPTR

MOV R1,#64H ;数据块长度 4+6\*16=100

LOOP1:

MOVX @DPTR,A ;写入数据

INC DPTR ;地址加一

DJNZ R1,LOOP1 ;判断是否传送完毕

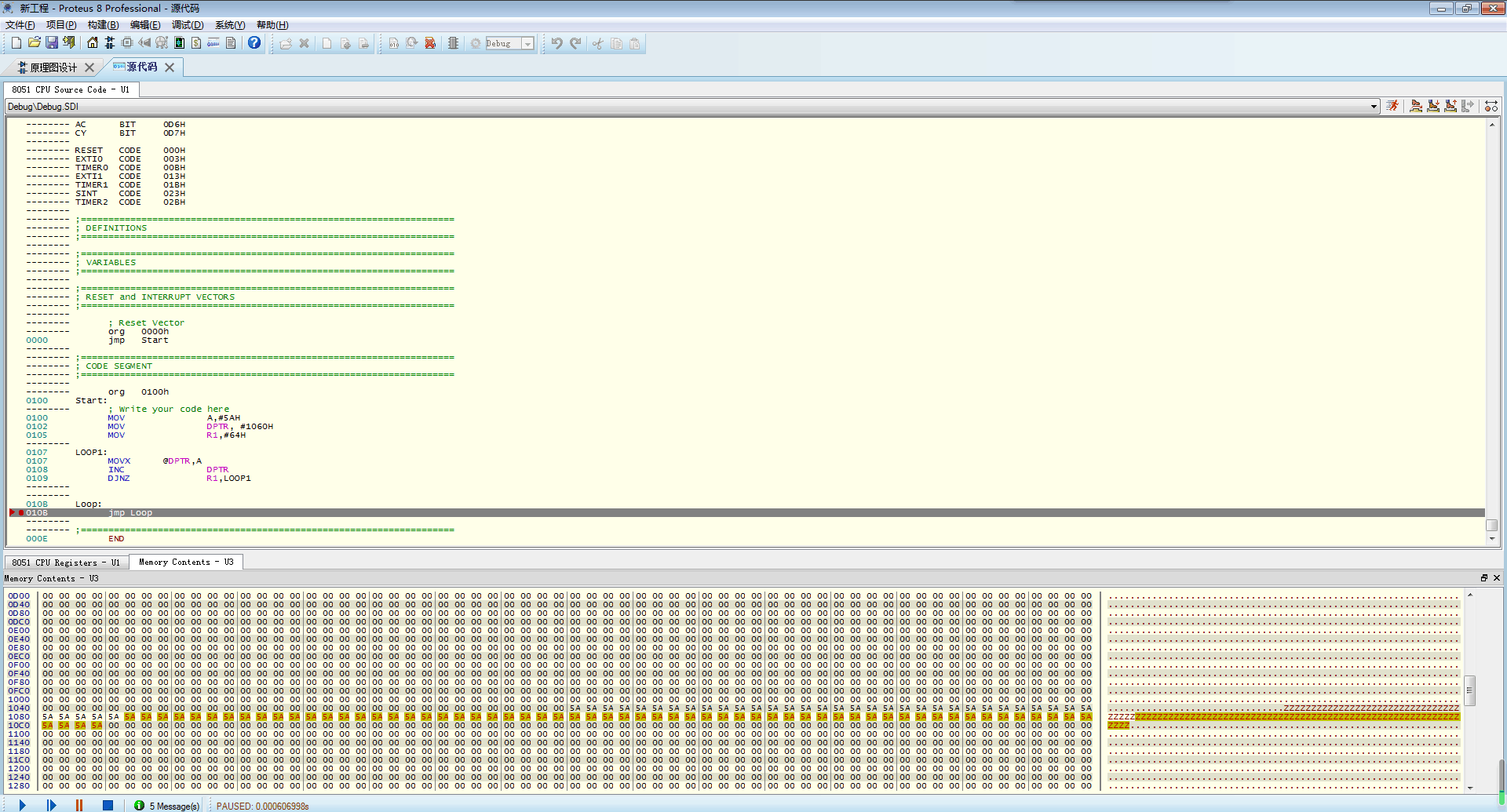


图3 实验步骤一截图

步骤2：将1060H地址开始的20个字节传送到30H开始的目的地址

代码如下：

MOV R1,#14H ;数据块长度 4+1\*16=20

MOV R0,#40H ;目的地址起始地址

MOV DPTR,#1060H ;源地址起始地址

MOV P2,#00H ;外部RAM目的地址高八位强制置0

LOOP2:

MOVX A,@DPTR ;从原地址取数据

MOVX @R0,A ;写数据

INC DPTR ;目的地址加一

INC R0 ;源地址加一

DJNZ R1,LOOP2 ;判断是否传送完毕

Loop:

jmp Loop

;=========================================================

END

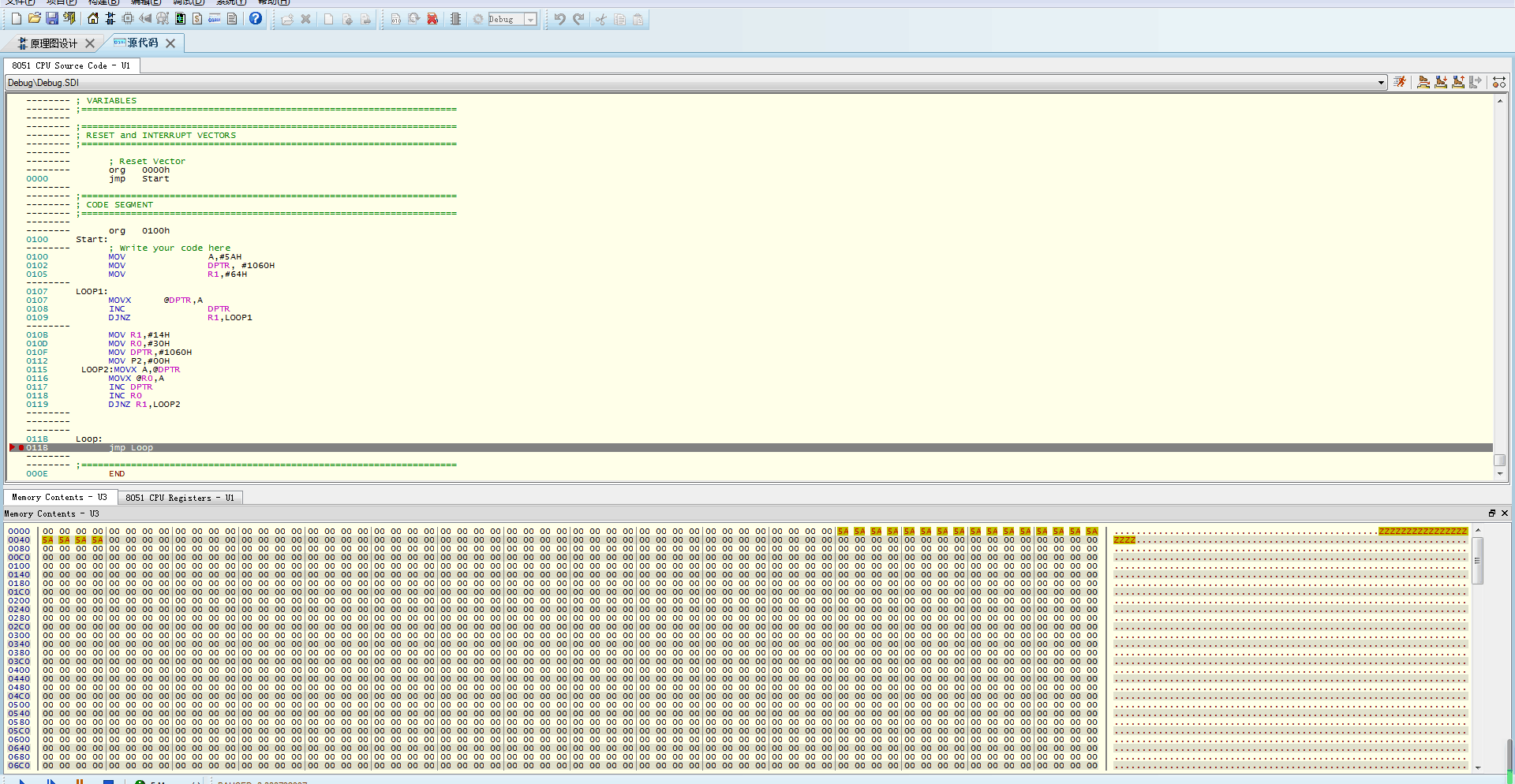


图4 实验步骤二截图

**4.2 实验结果分析**

用连续或单步方式运行程序，检查外部RAM中1060H和0040H地址中执行程序前后的内容变化，将实验结果截图记录。

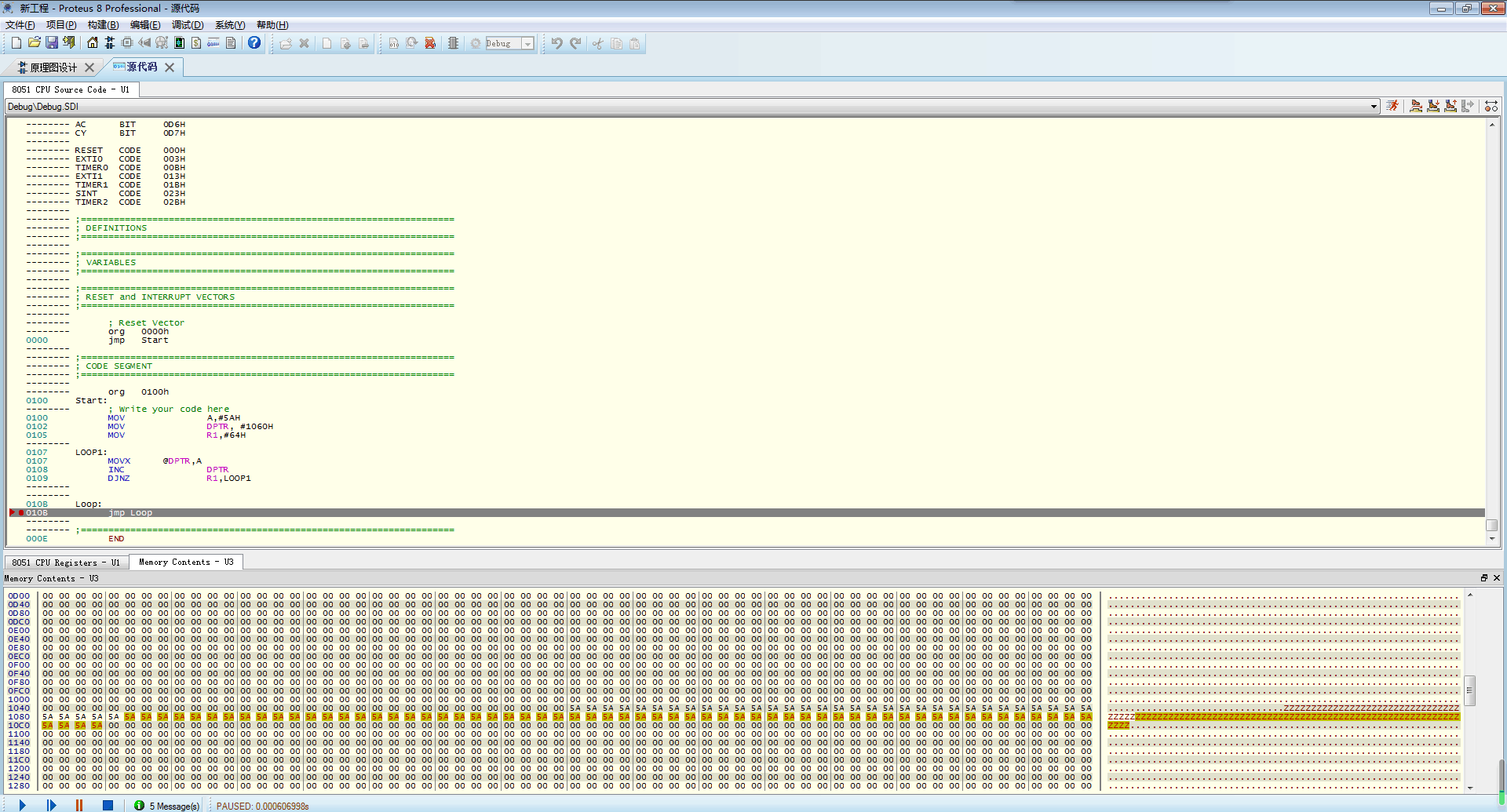


图5步骤1存储器观测结果

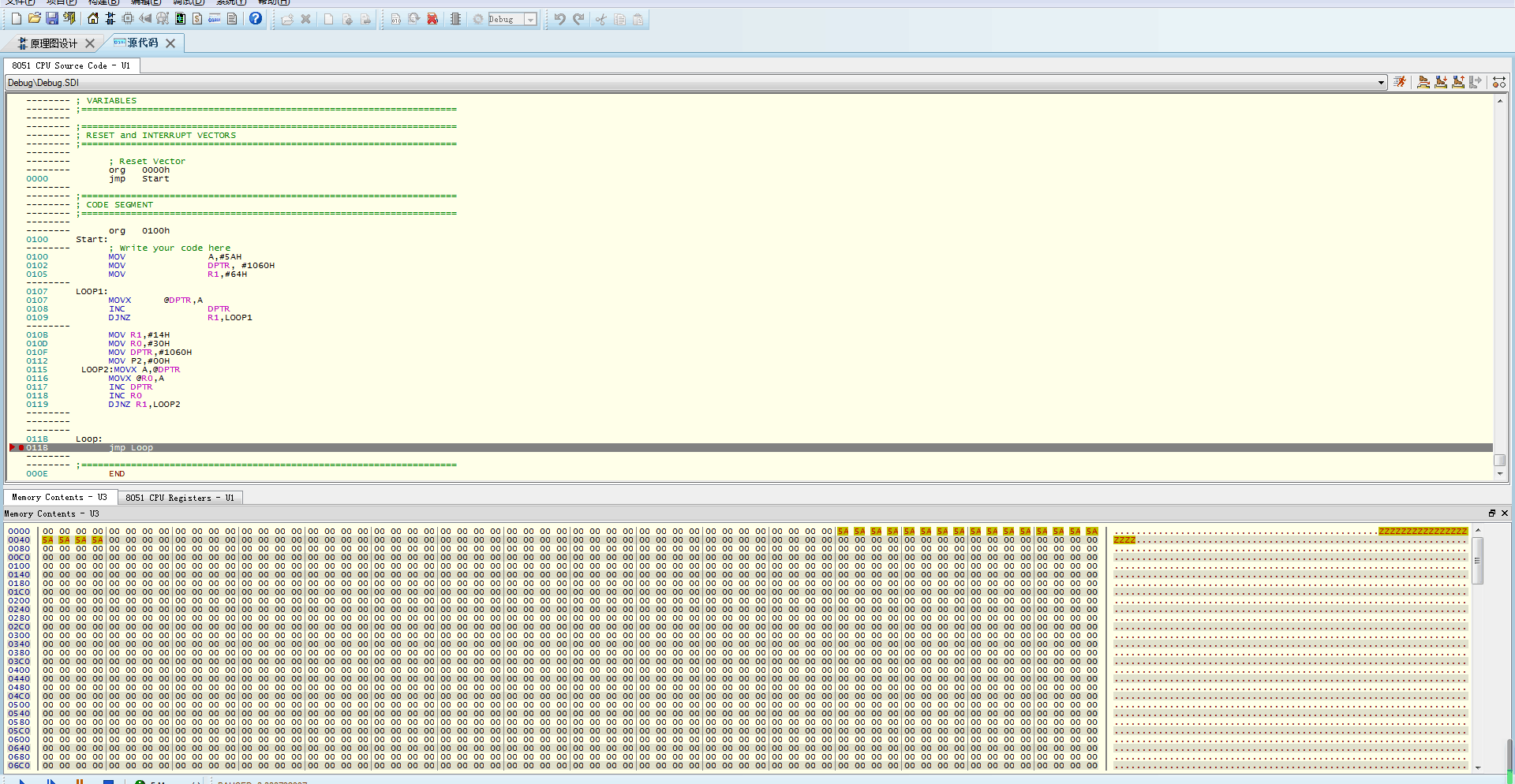


图6 步骤一存储器观测结果

五、思考问题

1. 简述MOVX两组指令，并说明其作用及区别。

【答】

第一组指令： MOVX @DPTR,A 累加器传送到外部RAM（16位）

MOVX @Ri, A 累加器传送到外部RAM（8位）

第二组指令： MOVX A,,@DPTR 外部RAM（16位地址）传送到A

MOVX A, @Ri 外部RAM（8位地址）传送到A

2. 单片机访问外部RAM时，使用了哪几个控制信号引脚？

【答】:片外数据存储器读选通控制输出；

:片外程序存储器读选通控制信号端；

：片外数据存储器写选通控制输出。

六、实验收获与建议

通过本次实验，对单片机的最小组成系统有了更深的理解，更加熟悉了通过汇编语言实现外部扩展RAM中的数据操作，激发了兴趣。

七、实验程序代码（程序需要完整、规范且有注释）

================================================

$NOMOD51

$INCLUDE (80C52.MCU)

;===============================================

; DEFINITIONS

;===============================================

;===============================================

; VARIABLES

;===============================================

;===============================================

; RESET and INTERRUPT VECTORS

;===============================================

; Reset Vector

org 0000h

jmp Start

;===============================================

; CODE SEGMENT

;===============================================

org 0100h

Start:

; Write your code here

MOV A,#5AH ;确定要附的值

MOV DPTR, #1060H ;起始地址送至DPTR

MOV R1,#64H ;数据块长度 4+6\*16=100

LOOP1:

MOVX @DPTR,A ;写入数据

INC DPTR ;地址加一

DJNZ R1,LOOP1 ;判断是否传送完毕

MOV R1,#14H ;数据块长度 4+1\*16=20

MOV R0,#40H ;目的地址起始地址

MOV DPTR,#1060H ;源地址起始地址

MOV P2,#00H ;外部RAM目的地址高八位强制置0

LOOP2:MOVX A,@DPTR ;从原地址取数据

MOVX @R0,A ;写数据

INC DPTR ;目的地址加一

INC R0 ;源地址加一

DJNZ R1,LOOP2 ;判断是否传送完毕

Loop:

jmp Loop

;=================================================================

END

## 项目2 二进制转ASCII 实验

一、实验目的

熟悉ASCII的编码原理，学习ASCII码与二进制之间的转换。

二、实验内容

使用AT89C52单片机对单片机内部RAM中存储的二进制数进行ASCII码转换操作，实验内容如下：

1. 将#9FH转为相应的ASCII；

2. 将结果存放到40H、41H地址；

3. 利用查表的方式实现。

三、实验原理及程序流程设计

**3.1 实验原理图功能及模块描述**

**1.实验原理详细说明**

利用汇编语言将#9FH转为对应ASCII码，再将结果存于40H、41H地址中。

**2 实验功能模块描述**

从左至右依次为置位电路，时钟振荡电路，8051单片机最小结构。

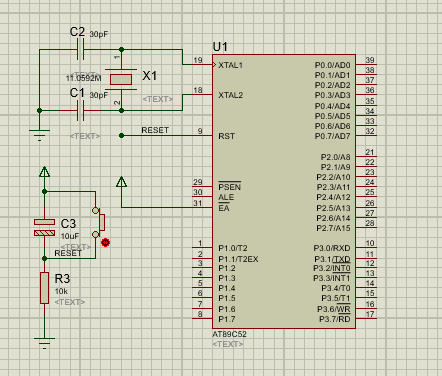


图1 实验原理图

**3.2 实验程序设计及流程图**

1. 实验程序设计思路：

首先，通过列表、查表找到#9FH对应的ASCII码，再将其按高字节和低字节分别储存在40H与41H地址中。

2. 实验流程图如2所示。



图2 程序流程图

四、实验步骤及结果分析

**4.1 实验步骤说明**

步骤1：将#9FH转为对应的ASCII码

步骤2：将其按低字节与高字节储存到40H与41H地址中。

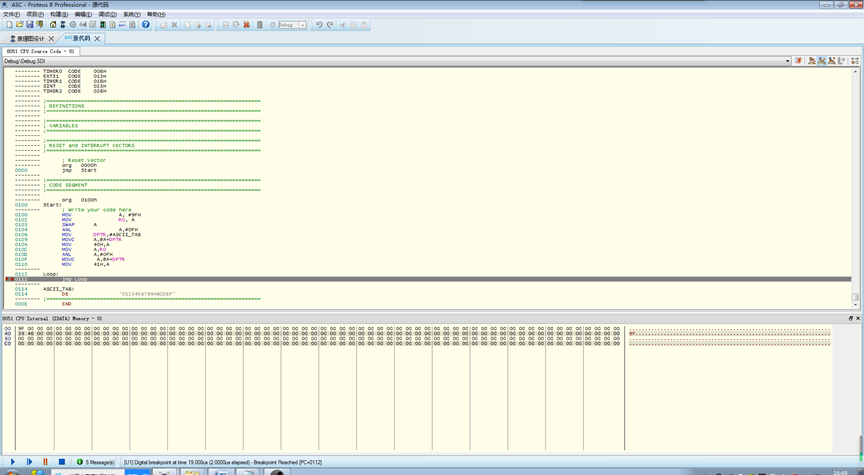


图3 实验步骤截图结果

**4.2 实验结果分析**

用连续或单步方式运行程序，检查40H-41H地址中执行程序前后的内容变化，将实验结果以及调试过程截图记录如下。

表格

描述已自动生成

图4 存储器观测结果

五、思考问题

1. 简述ASCII、BCD码与二进制之间的关系。

BCD(Binarycoded Decimal)码是二进制编码的十进制数，是用4位二进制码的组合代表十进制数的0，1，2，3，4，5，6 ，7，8，9十个数符。

ASCII码采用8位二进制代码对计算机使用的128个字符进行编，包括32个通用控制字符，10个阿拉伯数字，52个英文大，小字母，34个专用符号，其最高位添加一个“0”组成8位代码。

BCD码与ASCII码均是是二进制的一种应用，适用于计算机、单片机的运行与数据的传输。

2. 简述MOVC两条指令，并说明其作用及区别。

MOVC A, @A + DPTR 将程序存储器代码字节，传送到累加器

MOVC A, @A + PC 将程序存储器代码字节，传送到累加器

相同：均是查表指令，

区别：第一条指令可以给DPTR赋给任何一个16位的地址值，所以查表范围可达整个程序存储器64K字节空间的代码或常数，而第二条只能查这条指令所在地址以后256字节范围内的代码或常数。

六、实验收获与建议

通过该实验项目我了解了ASCII 的编码原理。学会了如何调用子函数，也熟悉了 MOVC指令和逻辑操作类指令，掌握了利用基址加变址寻址方式实现查表功能。

七、实验程序代码（程序需要完整、规范且有注释）

================================================

$NOMOD51

$INCLUDE (80C52.MCU)

;===============================================

; DEFINITIONS

;===============================================

;===============================================

; VARIABLES

;===============================================

;===============================================

; RESET and INTERRUPT VECTORS

;===============================================

; Reset Vector

org 0000h

jmp Start

;===============================================

; CODE SEGMENT

;===============================================

org 0100h

Start:

; Write your code here

MOV A, #9FH ;给累加器A赋值

MOV R0, A ;保存累加器A的内容到R0

SWAP A

ANL A,#0FH ;取A中值的高字节

MOV DPTR,#ASCII\_TAB ;表格首地址送入DPTR

MOVC A,@A+DPTR ;查高字节对应的ASCII码

MOV 40H,A ;保存高字节对应的ASCII码到40H

MOV A,R0 ;还原A

ANL A,#0FH ;取A中值的低字节

MOVC A,@A+DPTR ;查低字节对应的ASCII码

MOV 41H,A ;保存低字节对应的ASCII码到41H

RET

Loop:

jmp Loop

ASCII\_TAB:

DB '0123456789ABCDEF'

;=========================================================

END