

Министерство образования Республики Беларусь  
Учреждение образования Белорусский государственный университет  
информатики и радиоэлектроники

Кафедра ЭВМ

Отчет по лабораторной работе № 4  
"Программирование системного таймера"

Вариант 7

Выполнил:  
студент группы 950503  
Полховский А.Ф.

Проверил:  
Одинец Д. Н.

Минск 2021

## 1. Задание

Задание состоит из двух частей. Первая часть общая для всех. Вторая часть по вариантам. Варианты выдаются преподавателем после того, как студенты разделятся на бригады.

Первая часть (общее задание). Запрограммировать второй канал таймера таким образом, чтобы динамик компьютера издавал звуки.

Вторая часть (два варианта):

1. Для всех каналов таймера считать слово состояния и вывести его на экран в двоичной форме.
2. Для всех каналов таймера рассчитать коэффициент деления и вывести его на экран в шестнадцатеричной форме.

## 2. Теоретические сведения

Системный таймер - устройство, подключенное к линии запроса IRQ0 и вырабатывающее прерывание int 8h. Таймер реализуется на микросхеме Intel 8253 или 8254 (Отечественные аналоги: К1810ВИ53 и К1810ВИ54).

Таймер состоит из трёх независимых каналов: 0 - отсчитывает текущее время после включения компьютера, 1- для работы с контроллером прямого доступа к памяти, 2 – связан с системным динамиком. Каждый канал содержит регистры:

- состояния канала RS (8 разрядов);
- управляющего слова RSW (8 разрядов);
- буферный регистр OL (16 разрядов);
- регистр счетчика CE (16 разрядов);
- регистр констант пересчета CR (16 разрядов)

Каналы таймера подключаются к внешним устройствам при помощи линий:

- GATE - управляющий вход;
- CLOCK - вход тактовой частоты;
- OUT - выход таймера.

Регистр счетчика CE работает в режиме вычитания. Его содержимое уменьшается по спаду сигнала CLOCK при условии, что на вход GATE установлен уровень логической 1. В зависимости от режима работы таймера при достижении счетчиком CE нуля тем или иным образом изменяется выходной сигнал OUT. Буферный регистр OL предназначен для запоминания текущего содержимого регистра счетчика CE без остановки процесса счета. После запоминания буферный регистр доступен программе для чтения.

Регистр констант пересчета CR может загружаться в регистр счетчика, если это требуется в текущем режиме работы таймера. Возможны шесть режимов работы таймера, которые делятся на три типа:

Режимы 0, 4 - однократное выполнение функций.

Режимы 1, 5 - работа с перезапуском.

Режимы 2, 3 - работа с автозагрузкой.

Для подключения динамика используется порт 61h.

### 3. Листинг программы

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <dos.h>

unsigned int lab_notes[] = {
    392, 329, 329, 392, 329, 329, 392, 349, 329,
    392, 329, 329, 392, 329, 329, 392, 349, 329,
    392, 329, 329, 392, 329, 329, 392, 349, 329, 1
};

unsigned int ndelay[] = {
    400, 400, 400, 1000, 400, 400, 400, 800, 1000,
    400, 400, 400, 1000, 400, 400, 400, 800, 1000,
    400, 400, 400, 1000, 400, 400, 400, 800, 1000,
};

void Menu(void) {
    printf("\n1. Play sound");
    printf("\n2. Print channels kd");
    printf("\n3. Print state words");
    printf("\n5. Exit.");
    printf("\n\nEnter choice: ");
    return;
}

void MenuChoice(int *number) {
    for (; ; ) {
        while (scanf("%d", number) != 1) {
            printf("\nIncorrect input. Enter only digits: ");
            while (getchar() != '\n');
            continue;
        }
        if (*number != 1 && *number != 2 && *number != 3
            && *number != 4 && *number != 5 && *number != 6) {
            printf("\nIncorrect input. Choose one of points in menu: ");
            continue;
        }
        if (getchar() != '\n') {
            printf("\nIncorrect input. Enter only digits: ");
            while (getchar() != '\n');
            continue;
        }
        else {
            break;
        }
    }
    return;
}

void RandomBoardEnter(long *number) {
    printf("\nEnter random boarder: ");
    for (; ; ) {
        while (scanf("%ld", number) != 1) {
            printf("Incorrect input. Enter only digits: ");
            while (getchar() != '\n');
            continue;
        }
        if (*number <= 0 || *number >= 65535) {
```

```

        printf("Incorrect input. Enter right value (less then
65535): ");
        continue;
    }
    if (getchar() != '\n') {
        printf("Incorrect input. Enter only digits: ");
        while (getchar() != '\n');
        continue;
    }
    else {
        break;
    }
}
return;
}

void TurnSpeaker(int isActive) {
    if (isActive) {
        outp(0x61, inp(0x61) | 3);
        return;
    }
    else {
        outp(0x61, inp(0x61) & 0xFC);
        return;
    }
}

void SetSoundFreq(unsigned int freq) {
    long base = 1193180; //частоты генератора сигналов таймера
    long kd;
    outp(0x43, 0xB6); //10110110 канал 2, операция 4, режим 3, формат 0
    kd = base / freq;
    outp(0x42, kd % 256);
    kd /= 256;
    outp(0x42, kd);
    return;
}

void PlaySound() {
    int i;
    int c;
    printf ("1 - lab melody\n");
    scanf ("%d", &c);
    if (c==1)
    {
        for (i = 0; lab_notes[i] != 1; i++)
        {
            SetSoundFreq(lab_notes[i]);
            TurnSpeaker(1);
            delay(ndelay[i]);
            TurnSpeaker(0);
        }
    }

    if (c==2)
    {
        for (i = 0; godFather[i] != 1; i++)
        {
            SetSoundFreq(godFather[i]);
            TurnSpeaker(1);
            delay(godFatherdelay[i]);
            TurnSpeaker(0);
        }
    }
}

```

```

    }
}

void CharToBin(unsigned char state, char *str) {
    int i, j;
    char temp;
    for (i = 7; i >= 0; i--) {
        temp = state % 2;
        state /= 2;
        str[i] = temp + '0';
    }
    str[8] = '\\0';
}

void PrintKd() {
    int iChannel;
    long j;
    long kd_low, kd_high, kd, kd_max;

    for (iChannel = 0; iChannel < 3; iChannel++) {
        kd_max = 0;
        for (j = 0; j < 65535; j++) {
            switch (iChannel) {
                case 0: {
                    outp(0x43, 0x0);
                    kd_low = inp(0x40);
                    kd_high = inp(0x40);
                    kd = kd_high * 256 + kd_low;
                    break;
                }
                case 1: {
                    outp(0x43, 0x40);
                    kd_low = inp(0x41);
                    kd_high = inp(0x41);
                    kd = kd_high * 256 + kd_low;
                    break;
                }
                case 2: {
                    outp(0x43, 0x80);
                    kd_low = inp(0x42);
                    kd_high = inp(0x42);
                    kd = kd_high * 256 + kd_low;
                    break;
                }
            }
            if (kd_max < kd) {
                kd_max = kd;
            }
        }
        switch (iChannel) {
            case 0: {
                printf("Channel 0x40 kd: %ld\\n", kd_max);
                break;
            }
            case 1: {
                printf("Channel 0x41 kd: %ld\\n", kd_max);
                break;
            }
            case 2: {
                printf("Channel 0x42 kd: %ld\\n", kd_max);
                break;
            }
        }
    }
}

```

```

    }
    return;
}

void StateWords() {
    char *bin_state;
    int iChannel;
    unsigned char state;
    bin_state = (char *)calloc(9, sizeof(char));
    if (bin_state == NULL) {
        printf("Memory allocation error");
        exit(EXIT_FAILURE);
    }

    for (iChannel = 0; iChannel < 3; iChannel++) {
        switch (iChannel) {
            case 0: {
                outp(0x43, 0xE2);          //11100010 0x40
                state = inp(0x40);
                CharToBin(state, bin_state);
                printf("Channel 0x40 word: %s\n", bin_state);
                break;
            }
            case 1: {
                bin_state[0] = '\0';
                outp(0x43, 0xE4);          //11100100 0x41
                state = inp(0x41);
                CharToBin(state, bin_state);
                printf("Channel 0x41 word: %s\n", bin_state);
                break;
            }
            case 2: {
                bin_state[0] = '\0';
                outp(0x43, 0xE8);          //11101000 0x42
                state = inp(0x42);
                CharToBin(state, bin_state);
                printf("Channel 0x42 word: %s\n", bin_state);
                break;
            }
        }
    }
    free(bin_state);
    return;
}

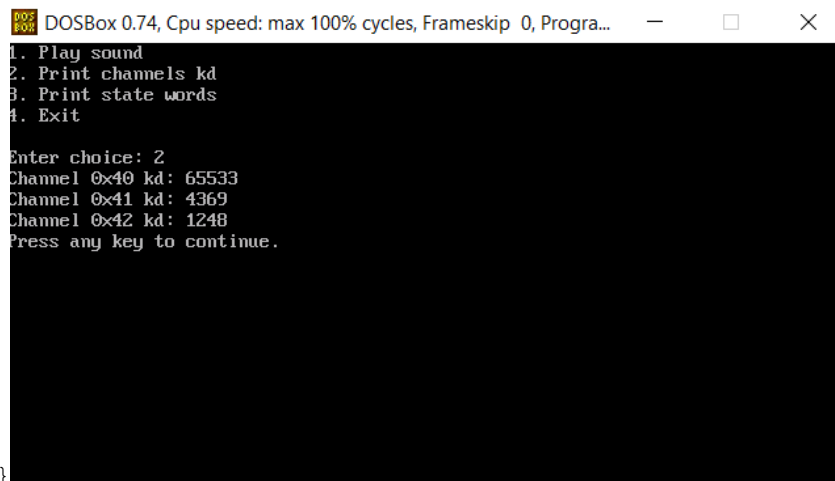
long RandomDigit() {
    long bl, bh;
    outp(0x43, 0x86); //4-5      00      6-7 10 2 порт
    bl = inp(0x42);
    bh = inp(0x42);
    bh *= 256;
    bh += bl;
    outp(0x61, inp(0x61) & 0xFC);
    return bh;
}

void RandTimerSet(long max) {
    outp(0x43, 0xB4); //10110100
    outp(0x42, max % 256);
    max /= 256;
    outp(0x42, max);
    outp(0x61, inp(0x61) | 1);
    return;
}

```

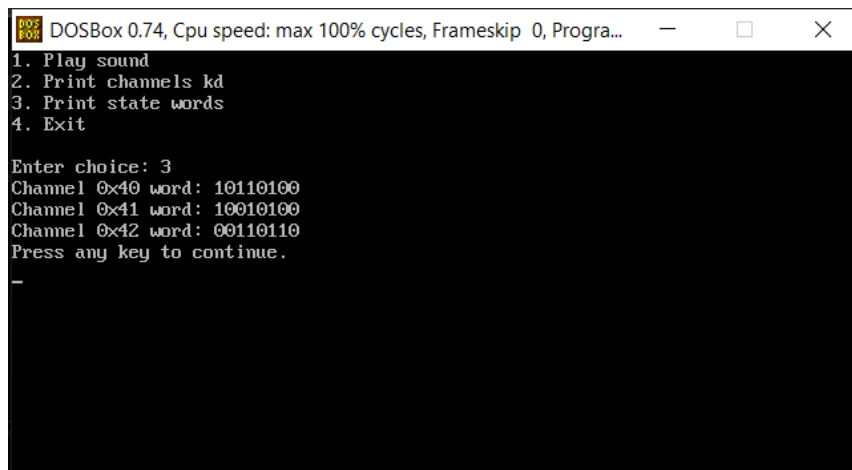
```
}
```

```
int main() {  
    int check_exit = 0;  
    system("cls");  
    Menu();  
    MenuChoice(&check_exit);  
    while (check_exit != 5) {  
        switch (check_exit) {  
            case 1: {  
                PlaySound();  
                break;  
            }  
            case 2: {  
                PrintKd();  
                break;  
            }  
            case 3: {  
                StateWords();  
                break;  
            }  
        }  
        check_exit = 0;  
        system("pause");  
        system("cls");  
        Menu();  
        MenuChoice(&check_exit);  
    }  
    return 0;  
}
```



DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Progra...

```
1. Play sound  
2. Print channels kd  
3. Print state words  
4. Exit  
  
Enter choice: 2  
Channel 0x40 kd: 65533  
Channel 0x41 kd: 4369  
Channel 0x42 kd: 1248  
Press any key to continue.
```



DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Progra...

```
1. Play sound
2. Print channels kd
3. Print state words
4. Exit

Enter choice: 3
Channel 0x40 word: 10110100
Channel 0x41 word: 10010100
Channel 0x42 word: 00110110
Press any key to continue.
-
```

**Вывод:** в ходе лабораторной работы было выполнено программирование системного таймера.