МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
КАФЕДРА АВТОМАТИЗОВАНОГО УПРАВЛІННЯ ТЕХНІЧНИМИ СИСТЕМАМИ

Лабораторна робота № 5

Варіант 1

По дисципліні «Програмування мікроконтролерних систем»

Тема: «Загальна система віддаленого управління з використанням шаблону кінцевого автомату.»

Виконали:

студенти групи ІТ-51

Бессмертний Р.С.

Цитовцева А.С.

_____

(підпис, дата)

Перевірив:

ст. викладач кафедри АУТС
Катін П. Ю.

_____

(підпис, дата)

Київ-2018

Мета: Розробка моделей, основ документації і прототипів СДУ рухомим об’єктом на основі обраного типу МК.

Хід роботи:

Варіант 1

Користуючись результатами лабораторної роботи 2, 3, 4, побудувати програму для рухомого об’єкту і ПДУ, що утворюють СДУ. У якості ПДУ обрати клавіатуру у вигляді 1 кнопки.

Для кнопки написати драйвер, що може рахувати кількість натискань кнопки і в залежності від цього числа реалізувати варіант команди. У якості контролера ДУ обрати плату налагодження будь якого типу для МК Cortex M3. Сигнал передавати по провідному каналу, користуючись *USART1*. Програма має передавати не менше 5 сигналів управління на віддалений об’єкт.

Віддалений об’єкт має відображати отримані команди і переходити у один з 5 можливих станів.

Написати і налагодити програми. Протестувати програми на реальному обладнанні, зробити тестування. При необхідності зробити корекцію у програмних і апаратних налаштуваннях.

1. Кінцевий автомат через switch.

```
switch(finite_state)
		{
			case STATE_GET_TIMER_INPUT:
				if (PPMBuffer != 0)
				{
					TM1637_display_all(PPMBuffer);
				}
				else
				{
					TM1637_display_all(TIMER_INPUT_IS_ZERO);
				}

				finite_state = STATE_IDLE;
				break;
		}
```

2. Кінцевий автомат через ООП

```
class IState {
	public:

		virtual void Execute() = 0;
		virtual IState * GetNextState() const = 0;
};

class StateContext {
	private: IState * _state;
```

```
        public:
                explicit StateContext(IState *initialState);

                void NextState(void);

                IState * GetState(void) const;

                void Execute() const;
};

class IActionState: public IState{
        public:
                virtual void Execute() = 0;
                virtual IState * GetNextState() const;
};

class ReadButton: public IState{
        private:
                IActionState * _action;
        public:
                virtual void Execute();
                virtual IState * GetNextState() const;
};

class GoForward: public IActionState{
        public:
                virtual void Execute();
};

class GoBackward: public IActionState{
        public:
                virtual void Execute();
};

class TurnLeft: public IActionState{
        public:
                virtual void Execute();
};

class TurnRight: public IActionState{
        public:
                virtual void Execute();
};

class Stop: public IActionState{
        public:
                virtual void Execute();
};
```

```
extern ReadButton readButton;
extern GoForward goForward;
extern GoBackward goBackward;
extern TurnLeft turnLeft;
extern TurnRight turnRight;
extern Stop stop;
```

Висновки: В даній лабораторній роботі код лабораторних робіт 1-4 був структурований в типовий шаблон finite state, що покращило його супродожуваність і читабельність.

https://github.com/roman-bessmertnyi/BUG-3000/tree/master/Bug-controller