Фазные напряжения выхода инвертора определяются следующим образом, (вид фазы А представлен на рис. 6)

$$uA_n := \left[\frac{2\cdot E\cdot(mA_n)}{3}\right] - \left(\frac{E\cdot mB_n}{3}\right) - \left(\frac{E\cdot mC_n}{3}\right)$$

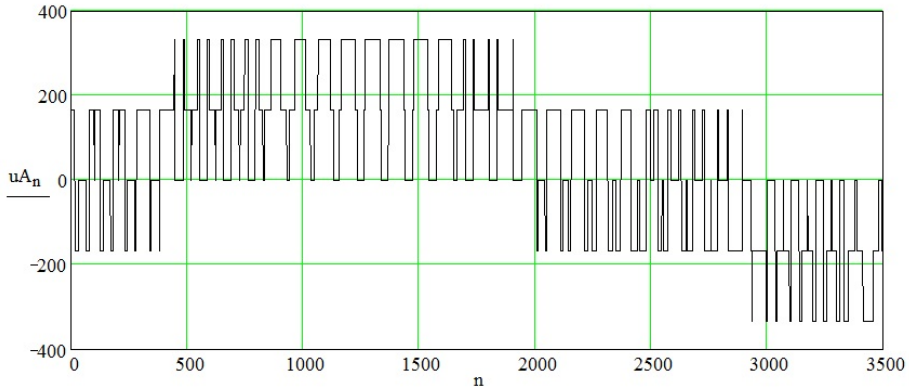

Рис. 6. Выходное напряжение фазы А

Построение подобных моделей позволяет построить выходную характеристику 3-х фазного инвертора в зависимости от коэффициента модуляции k_m, получить относительные длительности ширины управляющих импульсов для дальнейшего использования при физическом эксперименте, а также наглядно рассмотреть электромагнитные процессы в инверторе, при изменении входных данных (напряжение питания моста, значение k_m).

Литература

1. Мелешин В. И. Управление транзисторными преобразователями электроэнергии / В. И. Мелешин, Д. А. Овчинников – Москва: «Техносфера», 2011. – 576 с.

2. Чаплыгин Е. Е. Спектральное моделирование преобразователей с широтно-импульсной модуляцией. Учебное пособие по курсу «Моделирование электронных устройств и сиситем» / Е. Е. Чаплыгин – Москва: Изд-во МЭИ, 2009. – 56 с.

Д.Е. Яблоков

ИСПОЛЬЗОВАНИЕ ОБОБЩЕННЫХ КОНЦЕПЦИЙ В ОБЪЕКТНО-ОРИЕНТИРОВАННЫХ ЯЗЫКАХ ПРОГРАММИРОВАНИЯ

(Самарский государственный университет)

Сфера компьютерных наук, которая возникла сравнительно недавно, проделала стремительное развитие от состояния, когда в ней был занят лишь небольшой круг специалистов, до состояния, когда речь идет о повсеместном использовании компьютерных технологий и связанных с ними результатов исследований в той или иной предметной области. Быстрый рост этого направления

сопровождался и продолжает пополняться формирующимся каркасом понятий, определяющих логические рамки для описания объектов или семейств объектов, работа с которыми становится возможна в терминах используемого понятийного аппарата. В большинстве случаев это приводит к появлению абстракций, определяющих семантическое и синтаксическое поведение экземпляра объекта, а так же к появлению семейства концепций, содержащих набор требований дающих представление о свойствах и ограничениях которыми должна обладать абстракция, являющаяся моделью одной или нескольких концепций. Тип, который моделирует концепцию, обязан удовлетворять ее требованиям, включающим наборы допустимых выражений, ассоциированных типов, инвариантных характеристик времени исполнения и гарантий сложности. Наличие концепций очень важно, потому что большинство предположений о типах, используемых в обобщенном программировании, может быть выражено как в терминах соответствия концепциям, так и в терминах отношений между различными концепциями.

Обобщенная модель концепции однопроходного итератора.

```
public interface IIterator<I extends IIterator<I>> extends Cloneable
{
	void advance();
	I getIterator();
};
```

Обобщенная модель концепции итератора вывода.

```
public interface IOutputIterator<I extends IOutputIterator<I, E>, E> extends IIterator<I>
{
	IOutputIterator<I, E> clone();
	void put(E element);
};
```

Обобщенная модель концепции итератора ввода.

```
public interface IInputIterator<I extends IInputIterator<I, E>, E> extends IIterator<I>
{
	boolean equalTo(IInputIterator<I, E> other);
	IInputIterator<I, E> clone();
	E get();
};
```

Обобщенная модель концепции однонаправленного итератора.

```
public interface IForwardIterator<I extends IForwardIterator<I, E>, E> extends
IInputIterator<I, E>, IOutputIterator<I, E>
{
	IForwardIterator<I, E> clone();
};
```

Обобщенные модели концепций функциональных адаптеров.

1. Унарная функция

```
public interface IUnaryFunction<T, R>
{
	R execute(T argument);
};
```

2. Бинарная функция

```
public interface IBinaryFunction<T1, T2, R>
{
	R execute(T1 firstArgument, T2 secondArgument);
};
```

Обобщенные алгоритмы, не изменяющие набор данных.

Алгоритм FindIf:

```
public static <I extends IInputIterator<I, E>, E, UP extends IUnaryFunction<E, Boole-
an>>
I findIf(I first, I last, UP unaryPredicate)
{
	I iterator = first.clone().getIterator();
	while((!iterator.equalTo(last)) && (!unaryPredicate.execute(iterator.get())))
		iterator.advance();
	return iterator;
}
```

Алгоритм FindFirstOf:

```
public static <I1 extends IInputIterator<I1, E1>, I2 extends IForwardIterator<I2, E2>,
E1, E2, BP extends IBinaryFunction<E1, E2, Boolean>>
I1 findFirstOf(I1 first1, I1 last1, I2 first2, I2 last2, BP binaryPredicate)
{
	I1 iterator1 = first1.clone().getIterator();
	for(; !iterator1.equalTo(last1); iterator1.advance())
	{
		I2 iterator2 = first2.clone().getIterator();
		for(; !iterator2.equalTo(last2); iterator2.advance())
		{
			if(binaryPredicate.execute(iterator1.get(), iterator2.get()))
				return iterator1;
		}
	}
	return last1;
}
```

Примеры обобщенных алгоритмов, изменяющих набор данных.

Алгоритм copy:

```
public static <I1 extends IInputIterator<I1, E>, I2 extends IOutputIterator<I2, E>, E>
I2 copy(I1 first, I1 last, I2 result)
{
	I1 iterator1 = first.clone().getIterator();
	I2 iterator2 = result.clone().getIterator();
	for(; !iterator1.equalTo(last); iterator2.advance(), iterator1.advance())
		iterator2.put(iterator1.get());
	return iterator2;
}
```

Алгоритм copyBackward:

```
public static <I1 extends IBidirectionalIterator<I1, E>, I2 extends IBidirectionalIter-
ator<I2, E>, E>
I2 copyBackward(I1 first, I1 last, I2 result)
{
	I1 iterator1 = last.clone().getIterator();
	I2 iterator2 = result.clone().getIterator();
```

```
        while(!iterator1.equalTo(first))
        {
                iterator2.retreat();
                iterator1.retreat();
                iterator2.put(iterator1.get());
        }
        return iterator2;
}
```

Современные языки высокого уровня, в большинстве своем, поддерживают парадигму объектно-ориентированного программирования, использующую иерархии полиморфных типов данных, связанных отношением наследования. Это позволяет, применяя дополнительный уровень абстракции, предоставляемый парадигмой, ссылаться на значение полиморфного объекта, манипулировать им, не указывая точного типа, в рамках иерархии наследования. Обобщенное программирование, как развитие объектно-ориентированного направления, также определяет собственный уровень абстракции. Ключевая абстракция обобщенного программирования представляет собой набор абстрактных требований к типам, описываемых с помощью определения абстрактных видов, т.е. семейств абстрактных типов, моделирующих набор требований, который определяет интерфейс и семантическое поведение. Алгоритм, написанный в обобщенном стиле, может применяться для любых типов, удовлетворяющих синтаксическим и семантическим требованиям, которые он предъявляет к своим аргументам. Любой подобный алгоритм состоит из двух частей: конкретных инструкций, определяющих шаги исполнения (императивное управление и процедурно-операторный стиль), и набора абстрактных требований (определение абстрактных концепций), которым типы его аргументов должны точно соответствовать. Процедурно-операторный подход помогает нам создать конкретную специализацию алгоритма, делая акцент на обработке данных, т.е. на процессе необходимых вычислений. Основой для него служит предоставление механизмов передачи аргументов функциям и получение от них возвращаемого вычисленного значения. Парадигма обобщенного программирования предполагает использование перечня определенных и систематизированных требований к абстрактному типу и представляет собой концепции, описывающие свойства, которыми должен обладать тип, чтобы удовлетворять предъявляемым требованиям. Поддержка соответствующего стиля программирования осуществляется через специально предназначенные для этого языковые конструкции. Но важно не то, какими средствами обладает язык, а то, что этих средств, а также поддерживаемых языком концепций, принципов и идиом достаточно для реальной поддержки одной или нескольких парадигм программирования.

Обобщенные концепции – это мощный инструмент решения задач, используемый для описания поведения и отношений между дискретными объектами и их семействами. Практические задачи могут быть смоделированы в виде абстракций для различных областей научные вычислений или исследований. Достоинством таких абстракций является тот факт, что найденное решение

проблемы, в случае грамотно и правильно построенной обобщенной концепции, может быть использовано для решения подобных проблем в широком диапазоне иных областей. При сосредоточении на сути этих проблем, а именно на концепциях описывающих синтаксические и семантические особенности конкретных объектов и их отношений специалисты могут найти решение не только для отдельных проблем, но для целых классов задач, решаемых в ходе научной работы.

Литература

1. Вандевурд Д., Джосаттис Н. М. Шаблоны C++: справочник разработчика. – М.: Вильямс, 2003. – 544 с.

2. Саттер Г., Александреску А. Стандарты программирования на C++. – М.: Вильямс, 2005. – 224 с.

3. Степанов А., Пол М.-Д. Начала программирования. – М.: Вильямс, 2011. – 272 с.

4. Страуструп Б. Язык программирования C++. Специальное издание. – М.: Издательство Бином, 2011 г. – 1136 с.

5. Яблоков Д. Применение парадигмы обобщенного программирования в объектно-ориентированных языках. Информатика, моделирование, автоматизация проектирования: сборник рекомендованных научных трудов / под ред. Н. Н. Войта. – Ульяновск : УлГТУ, 2013. – с. 113-118.

А.М. Егоров

ПЕРСПЕКТИВНАЯ ОТКАЗОУСТОЙЧИВАЯ БОРТОВАЯ СИСТЕМА УПРАВЛЕНИЯ ТИПОВЫМ НАНОСПУТНИКОМ

(Самарский государственный аэрокосмический университет имени академика С.П. Королева (национальный исследовательский университет))

Современное направление развития сферы космических экспериментов, проводимых университетами мира, неразрывно связано с использованием наноспутников. Аппараты широко применяемого стандарта CubeSat характеризуются малой массой до 3 кг, малыми габаритными размерами: от одного до трёх кубических блоков с гранью 10 см.

Широкое использование наноспутников во многом обусловлено малым временем их разработки, а также значительной дешевизной запуска. Такие аппараты могут выводиться на орбиту попутно, в добавление к основной полезной нагрузке.

При разработке наноспутников инженеры стремятся использовать датчики и исполнительные механизмы, имеющие малые массогабаритные характеристики. Такой подход позволяет представить типовой набор элементов и систем борта, присущий большинству разрабатываемых аппаратов.