Выбор платформы для реализации систем управления

Аннотация. Данная статья носит учебно–методический характер и ориентирована на студентов старших курсов специальностей, связанных с системной инженерией и/или системами управления. В статье рассматриваются и сравниваются различные особенности инструментов для реализации готовых алгоритмов управления с позиции применения аппаратных, аппаратно-программных или программных инструментов. Как результат, предлагается типовая архитектура программно – аппаратной платформы для размещения на ней большинства современных технических систем управления.

ОГЛАВЛЕНИЕ

Типичная архитектура системы управления

Сегодня большинство объектов и систем управления относятся к классу распределенных объектов и систем, что подразумевает определенную пространственную протяженность и наличие многих локальных подсистем в составе системы управления такими объектами. В общем виде, такой распределенный объект управления и соответствующую ему систему управления можно представить с помощью рисунка 1.

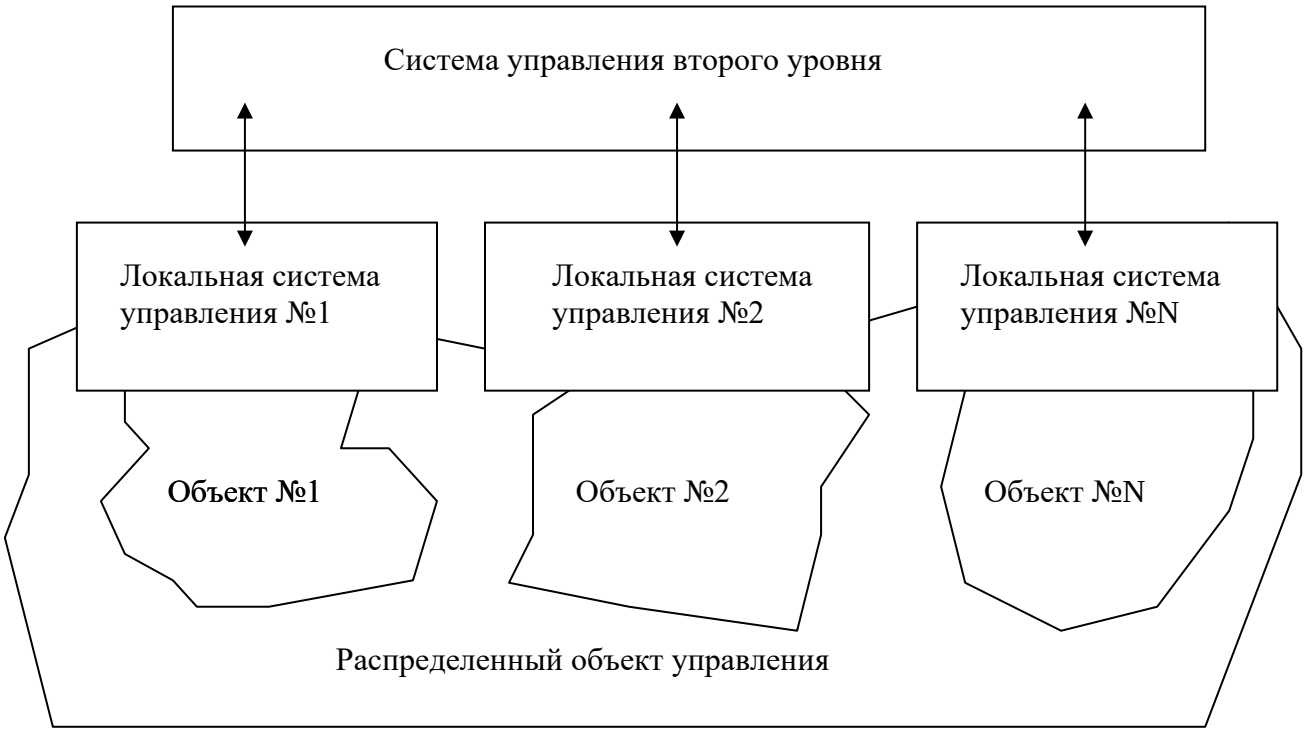

Рис 1. Распределенный объект и соответствующая ему система управления.

Рассмотрим более внимательно локальную систему управления. Условно такие локальные системы можно разделить на два подкласса: это системы, реализованные аппаратными средствами и системы, реализованные программными средствами:

- Аппаратные системы. К ним относятся системы, основная модель управления которых выполнена средствами механики, пневматики, гидравлики, электроники или их комбинациями.
- Программные системы. К ним относятся системы, основная модель управления которых выполнена с помощью универсальных средств обработки цифровой информации. Как правило, такими средствами являются микроконтроллеры или микро компьютеры. При этом задачи, связанные с преобразованием физической величины в числовую величину и обратно, вынесены в отдельные функциональные блоки АЦП (аналого-цифровой преобразователь) и ЦАП (цифроаналоговый преобразователь).

Соответствующие структуры названных систем можно представить в следующем виде (рисунок 2)

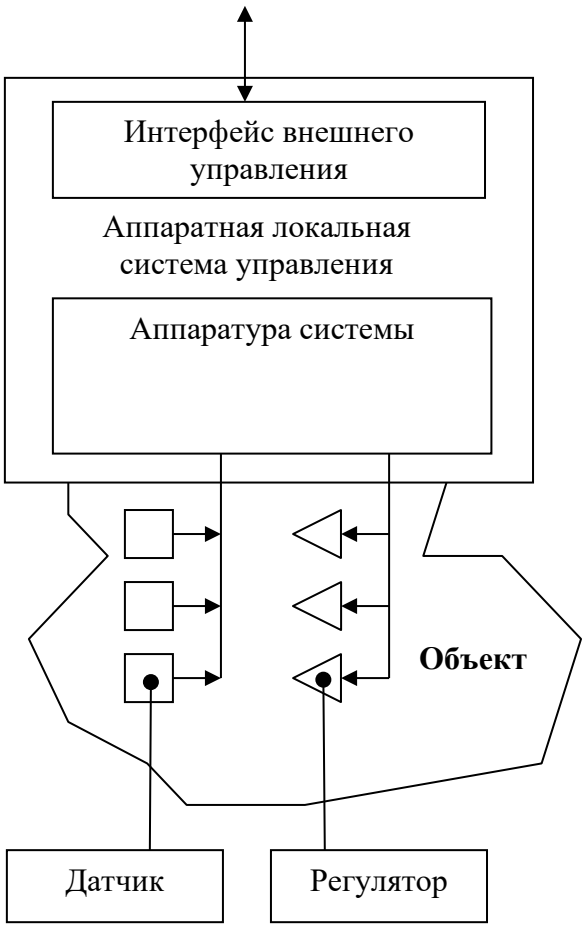

Рис 2а. Аппаратная система управления

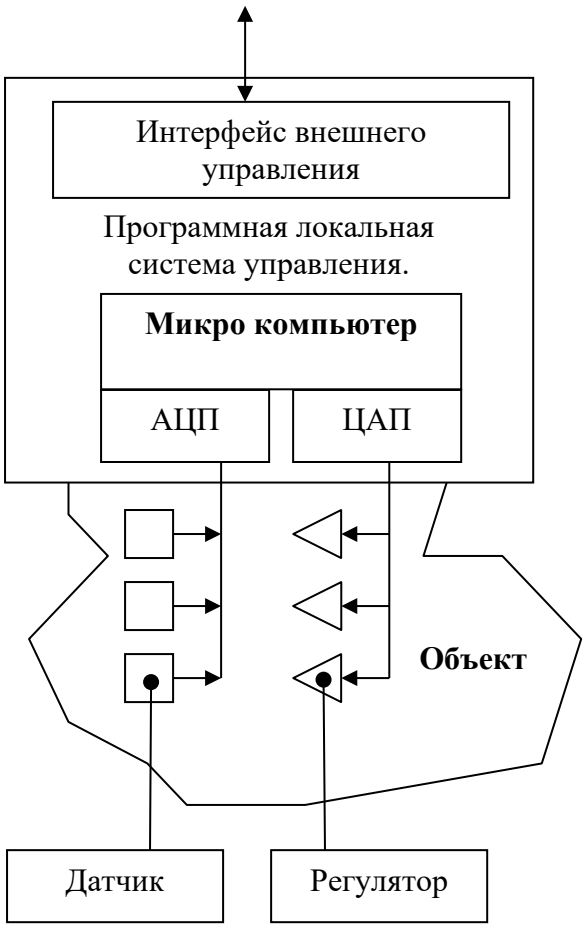

Рис 2б. Программная система управления.

В любом случае, основными задачами управления являются задачи, которые формализуются некоторым интегральным параметром системы, именуемым как состояние системы или “S” . Например, такое состояние можно определить как некоторую функцию “F”, заданную через список ее параметров “p”, то есть, показаний датчиков.

$$S = F(p1, p2, \cdots pN)$$

В числе таких задач можно назвать:

- Устойчивую стабилизацию значения состояния “S” в некоторой заданной точке;
- Максимально быстрый переход из одного состояние в другое;
- Переход между двумя состояниями по наперед заданной траектории;
- Различные сочетания задач сформулированных выше.

В любом случае, качество выполняемых задач зависит от точности и быстродействия первичных преобразователей (датчиков) и регуляторов, а также эффективности и быстродействия модели управления. Под эффективностью будем понимать наилучшее соответствие модели поставленной для нее задаче. Как правило, такое соответствие неразрывно связано с точностью и скоростью реализации функции “F”. Причем точность и скорость, как правило, прямо связаны с количеством анализируемых параметров и ее аналитической сложностью

Опираясь на приведенные выше качественные определения, сравним аппаратную и программную реализации по нескольким ключевым характеристикам. При этом будем считать, что датчики и регуляторы являются общими для реализаций стандартными покупными изделиями.

Сравнение аппаратной и программной реализаций систем управления

Сравнение реализаций по оценкам параметров жизненного цикла систем

Значительную роль в жизненном цикле системы имеют этапы ее разработки, а также эксплуатации и сопровождения. Этапы разработки и сопровождения имеют ряд схожих особенностей. В первую очередь, это разработка или совершенствование управляющей модели и ее взаимодействия с внешней средой. Если создание управляющей модели это начало жизненного цикла, то совершенствование в процессе эксплуатации, это продление времени жизненного цикла.

Сравним преимущества аппаратной и программной реализации при создании или совершенствовании модели управления.

Табл. 1

Создание или совершенствование системы модели управления	
Аппаратная реализация системы	Программная реализация системы
1. Теоретическая разработка модели. 2. Разработка схемотехники. 3. Закупка аппаратных комплектующих, изготовление и отладка макета.	1. Теоретическая разработка модели. 2. Разработка компьютерной программы. 3. Отладка программы в типовой архитектуре аппаратной среды.
Вывод №1: Существенное преимущество программной реализации за счет отсутствия при отладке компьютерной программы времени и затрат на изготовление (или исправлений) макета, а также за счет более высокой точности аналитических преобразований.	
4. Разработка конструкторской документации на изготовление. 5. Закупка аппаратных комплектующих, изготовление и отладка каждого серийного образца.	4. Отсутствует 5. Закупка и интеграция стандартных аппаратных узлов (АЦП, ЦАП, Миро компьютера). Копирование программы со стенда на рабочий образец.

Вывод №2: Существенное преимущество программной реализации за счет низкой стоимости и времени интеграции аппаратных средств и программ в образец системы. В первую очередь, это обусловлено отсутствием дорогостоящих и долговременных этапов разработки конструкторской документации и изготовления образцов системы, а также их индивидуальной настройки.

Обзор рынка 2017 года для наиболее популярных микро компьютеров, АЦП и ЦАП, позволяет получить следующие индикативные показатели, которые представлены в таблицах 2 и 3.

Табл. 2

Обзор рынка 2017 года для наиболее популярных микро компьютеров								
Микро компьютер	Число ядер	Разрядность	Рабочая частота	Оперативная память	Встроенная Flash-память	Интерфейсы	Число доступных ОС	Типичная стоимость в US $
Banana Pi M64/BPI-M64	4	**64**	1,2GHz	**2GB DDR3** SDRAM 733MHz	8GB eMMC или16, 32, 64GB	**Ethernet10-1000Mb/s**, WiFi, Bluetooth,**USB**, **GPIO:**(UART,I2C,SPI)	**7**	$48
Orange PI PC	4	32	1,2GHz **1,6GHz**	**1GB DDR3**	microSD до 64GB	**Ethernet10-100Mb/s**, **USB**, **GPIO:**(UART,I2C,SPI)	**23**	$19 - $24
Raspberry Pi 3 Model B	4	**64**	1,2GHz	**1GB** SDRAM LPDDR2 (900 MHz)	слот для карты памяти MicroSDHC, USB Boot Mode	**Ethernet10-1000Mb/s**, WiFi, Bluetooth,**USB**, **GPIO:**(UART,I2C,SPI)	**17**	$35

Примечание. Список типичных ОС обычно включает: Ubuntu, Debian, Fedora, Arch Linux, Gentoo, RISC OS, Android, Firefox OS, NetBSD, FreeBSD, Slackware, Tiny Core Linux, Windows 10 IOT

Основными выводами, которые следуют из таблицы 2, являются следующие выводы:

- Благодаря использованию в микро компьютерах операционных систем (ОС), разработку, отладку и тестирование программного обеспечения модели управления и вспомогательного программного обеспечения можно выполнять в той же программно – аппаратной среде, что будет использоваться при эксплуатации.

- Стоимость закупки микро компьютера существенно меньше стоимости закупки и разработки оборудования для аппаратной реализации систем управления при аналогичной функциональной сложности систем.

Табл. 3

Обзор рынка 2017 года для наиболее популярных АЦП / ЦАП в Украине							
МОДУЛЬ	Разрядность АЦП/ЦАП	Частота измерений	Интерфейс подключения к микро компьютеру	Основное применение	ADC	DAC	Типичная стоимость в грн.
АЦП/ЦАП	**24/16**	**30 kHz**	**GPIO,** **GPIO: (I2C)**	АЦП/ЦАП плата расширения для Raspberry Pi (AD/DA)	ADS1256 - 24х битный АЦП высокой точности на 8 каналов (4 дифференциальных входа)	DAC8532 - 16ти битный, высокоточный, двухканальный ЦАП	963 грн.
АЦП/ЦАП PCF8591	**8/8**	**I2C**	**GPIO: (I2C)**	Общее применение	PCF8591	PCF8591	53 грн.

Сегодня мировой рынок АЦП/ЦАП является весьма обширным. По этой причине мы ограничимся типичными АЦП/ЦАП, применяемыми в Украине в 2017 году и отражающими граничные точности и стоимости.

Основными выводами, которые следуют из таблицы 2, являются следующие выводы:

- Точность 24 бита соответствует приведенной относительной погрешности 0,000006 процента, что значительно перекрывает потребности большинства современных систем управления. Точность 8 бит соответствует погрешности 0,4 процента, что вполне приемлемо для многих задач стабилизации физических параметров объектов.
- Стоимость модулей АЦП/ЦАП для большинства современных систем управления можно рассматривать как пренебрежительно малую.

Сравнение реализаций по оценкам наблюдаемости системы

Важнейшей свойством на всех этапах жизненного цикла системы управления является также свойство наблюдаемости, причем наблюдаемости, как отдельных параметров, так и различных состояний системы. Наблюдаемость параметров и состояний необходима *как* при отладке (в том числе и исправлении ошибок), при аттестации системы, а также при подстройке характеристик системы в процессе эксплуатации.
Сравним преимущества аппаратной и программной реализации в части наблюдаемости системы управления.

Поскольку все системы управления (особенно реализации их моделей управления) защищаются от посторонних воздействий путем их размещения в закрытом корпусе или другой защищающей оболочке, то основные возможности наблюдения реализуются через интерфейс внешнего управления (наблюдения). Сегодня на таком интерфейсе практически не встречаются различные патрубки или фитинги для присоединения устройств прямого измерения параметров пневматики или гидравлики. Вместо таких устройств в составе системы обычно используются преобразователи физической величины в электрический сигнал, а на интерфейс (обычно это электрический разъем) выводится очередной измеряемый или управляющий сигнал.

Вывод №3. Вполне очевидно, что при аппаратной реализации, количество контрольных точек наблюдения (сигналов) будет ограничено электрическим разъемом. В этом смысле, программная реализация имеет существенное преимущество, поскольку через стандартные интерфейсы микро компьютера (Ethernet, WiFi, Bluetooth,USB) возможно обеспечить доступ на уровне отдельных ячеек оперативной или встроенной (файловая система) памяти.

Сравнение реализаций по оценкам быстродействия

Еще одним важнейшим параметром систем управления является быстродействие. Исходными данными для обоснования того или иного требования к быстродействию системы управления, является динамика физических процессов в объекте управления. Если вынести за скобки быстродействие первичных преобразователей (датчиков и регуляторов), а также затраты на разработку, то для сравнения следует рассматривать скорость обработки сигналов при аппаратной реализации со скоростью АЦП/ЦАП преобразований и выполнения инструкций в микро компьютерах при программной реализации.

Вначале рассмотрим программную реализацию. При такой реализации наблюдение за некоторым процессом представимо с помощью потока цифровых значений, которые отображают мгновенные значения этого процесса в конкретные моменты времени. В этом случае, в качестве инструмента для получения оценки необходимого быстродействия, обычно применяется теорема Котельникова (Найквиста — Шеннона). Данная теорема доказывает, что для получения достоверной информации о некотором процессе частота измерений процесса должна быть как минимум в два раза выше, чем частота высшей (то есть, последней значимой) гармоники в спектре сигнала, отражающего этот процесс.

$$f_{измерений} \geq 2 \cdot f_{высшей\ гармоники.}$$

Оценку высшей (или последней значимой) гармоники можно получить, исходя из наперед заданной погрешности, которая возникает при отбрасывании гармоник с более высокой частотой, чем частота последней значимой гармоники.

Представим сигнал *f(x)*, который отображает процесс на некотором интервале времени (*T*), в виде дискретного разложения Фурье:

$$f(t) = \sum_{m=0}^{N} (b_m \cdot cos(m \cdot \omega \cdot t) + a_m \cdot sin(m \cdot \omega \cdot t))$$

где: $\omega = 2 \cdot \pi / T, \quad T = (t_{конца} - t_{начала})$ *или период первой гармоники.*

После того, как мы вычислим коэффициенты a_m и b_m:

$$b_m = \frac{2}{T} \int_{t_{начала}}^{t_{конца}} f(t) \cdot cos(m \cdot \omega \cdot t) \cdot dt$$

$$a_m = \frac{2}{T} \int_{t_{начала}}^{t_{конца}} f(t) \cdot sin(m \cdot \omega \cdot t) \cdot dt$$

у нас появляется возможность получить интенсивность каждой его гармоники (i_m) и энергетическую оценку всего или части спектра (*I*):

$$i_m = \sqrt{\frac{1}{2}(a_m^2 + b_m^2)}$$

$$I_0^N = \sqrt{\sum_{m=0}^{N} i_m^2}$$

Таким образом, через энергетическую оценку (сумму энергий гармоник) можно выразить погрешность отбрасываемой части спектра:

$$\gamma = \frac{I_K^N}{I_0^N} = \frac{\sqrt{\sum_{m=K}^{N} i_m^2}}{\sqrt{\sum_{m=0}^{N} i_m^2}} \quad \textit{где } K, \textit{ это индекс высшей гармоники}$$

Если выбрать некоторое значение погрешности, то можно определить индекс (*K*), а вслед за ним и частоту высшей гармоники:

$$I_K^N = \gamma \cdot I_0^N$$

$$\textit{Частота высшей гармоники} = \frac{K}{T}$$

Соответственно с теоремой Котельникова:

$$\textit{Частота измерений} \geq \frac{2 \cdot K}{T}$$

Поскольку обработка информации при программной реализации систем начинается с аналого-цифровых преобразований, приведем некоторые данные по быстродействию современных АЦП/ЦАП.

Табл. 4

Оценки быстродействия современных АЦП/ЦАП			
Частота измерений (преобразований)	Типичный алгоритм преобразования	Точность в битах	Реализация
Выше 4MHz	Специальные алгоритмы, как правило, на сверхбыстром суммировании токов или изучений	Как правило, не выше 8 бит	Специальные решения и (или) специальные микросхемы
От 0Hz до 4MHz	Алгоритмы суммирования токов, как правило, на базе резистивных сеток R-2R	От 14 до 8 бит. Типично 12 бит	Микросхемы от широкого круга производителей
От 0Hz до 60KHz	Сигма – дельта алгоритмы	От 24 до 12 бит	Микросхемы от широкого круга производителей

Наиболее широко на современном рынке микросхем АЦП/ЦАП представлены решения для частотных диапазонов от 0Hz до 4MHz с точностью 12 бит, что соответствует приведенной относительной ошибке 0,024 процента. Как следствие, такие преобразователи могут успешно применяться в очень широком диапазоне технических систем, выполняя до 4 миллионов измерений в секунду и обеспечивая достоверную обработку сигналов с высшей гармоникой до 2MHz

Скорость обработки информации микро компьютерами можно оценить по числу тактов, которые необходимы для выполнения командных инструкций. Число таких тактов существенно зависит от типа выполняемой операции. Наиболее быстро выполняются операции пересылки данных между внутренними регистрами процессора или записи констант во внутренние регистры. Как правило, в зависимости от архитектуры процессора, такие операции занимают несколько тактов. Команды суммирования, вычитания, умножения и деления зависят от типа операндов их размещения (регистры или оперативная память) и могут использовать достаточно большое число тактов, вплоть нескольких десятков тактов. В таблице 2 мы уже приводили тактовую частоту процессоров в современных микро компьютерах (1,2 – 1,6 GHz). Соответственно длительность тактов в таких процессорах будет составлять (1,2 – 1,6) наносекунды. Если рассматривать типичный временной диапазон выполнение названных выше команд, то он приблизительно составит от 8 до 32 наносекунд. Однако, чаще оценку скорости определяют с помощью flops. Flops это внесистемная единица, используемая для измерения производительности компьютеров, показывает сколько операций с плавающей запятой в секунду выполняет данная вычислительная система. Например, для микро компьютеров, приведенных в таблице 2, этот статистически усредненный параметр, составляет от 450 до 500 мега flops или 450 – 500 миллионов операций в секунду.

Вывод №4. Периодичность расчета воздействия управляющей системы на объект управления (или время расчета отклика системы управления) будет зависеть от двух параметров, то есть, от параметра flops и количества операций в алгоритме управления, необходимых для расчета управляющего воздействия.

Для ориентировочной оценки приведем несколько примеров:

Табл. 5

Быстродействие модели управления			
Относительная сложность алгоритма управления	Число операций в алгоритме управления	MFLOPS (мега flops) 4 ядра	Время расчета отклика системы в микро секундах
Простейшие алгоритмы	100 - 500	500	от 0,2 до 1
Алгоритмы средней сложности	500-5000	500	от 1 до 10
Алгоритмы большой сложности	5000 -50000	500	от 10 до 100

Вывод №5. Поскольку время измерения и время расчета отклика суммируются, то для типичной технической системы (время преобразования АЦП/ЦАП 4MHz составляет 0,25 микро секунды плюс время выполнения алгоритма средней сложности 10 микро секунд), полное время отклика составляет приблизительно 10 микро секунд, что соответствует частоте 100KHz. При применении простых алгоритмов управления полное время отклика составит порядка одной микросекунды, что соответствует частоте 1MHz. Эти оценки позволяют очертить класс объектов управления, которые эффективно могут управляться с помощью программной реализацией системы управления. К таким объектам можно отнести объекты, высшие гармоники которых располагаются в диапазоне от 500KHz до 50KHz.

Естественно возникает вопрос – возможно ли увеличить показатели быстродействия для систем с программной реализацией? Ответ на этот вопрос дают комбинированные подходы или программно – аппаратные реализации. Основой для таких реализаций сегодня являются программируемые логические интегральные схемы – ПЛИС (англ. programmable logic device, PLD). Особенностью таких интегральных схем является большое число логических элементов, связи между которыми можно построить и закрепить программным путем. Параллельный во времени характер работы логических элементов в составе ПЛИС позволяет создавать специальные параллельные алгоритмы системы управления, отклик которых составляет несколько тактов. Обзор и сравнительный анализ таких специальных алгоритмов далеко выходит за рамки нашего рассмотрения. Однако следует подчеркнуть, что сегодня тактовая частота современных ПЛИС приближается к значениям 1GHz, а следовательно, можно в качестве приблизительной оценки времени расчета оклика принять значения в диапазоне от 10 до нескольких десятков нано секунд. Очевидно, что при таких реализациях систем потребуются так же специальные реализации АЦП/ЦАП преобразования. Кроме того, поскольку при программно – аппаратной реализации уже невозможно провести четкую грань между программно – аппаратной и аппаратной реализацией, скорость отклика будет тесно связана со скоростью отклика простейших электронных компонентов. Сегодня успехи в области нано электронике позволяют делать оптимистические прогнозы в отношении разработки систем со сверхбыстрыми откликами.

Относительно свежий обзор программируемых логических интегральных схем можно посмотреть по ссылке:

http://category.alldatasheet.com/index.jsp?sSearchword=Fpga&sPage=1

Вывод №6. В заключение сравнения систем с аппаратной и программной реализацией отметим следующее:

- Сегодня элементами для аппаратной реализации типичных технических систем управления являются микросхемы операционных усилителей. Время отклика

типичных операционных усилителей на изменение входного сигнала, обычно находится в диапазоне от 10 до 100 наносекунд. При этом важно понимать, что для реализации простейшего алгоритма управления используется последовательное каскадное включение, как правило, не менее двух операционных усилителей, что сравнимо со временем отклика простейших алгоритмов для программной реализации.

- Как мы уже замечали в программно – аппаратной реализации, время отклика тесно связано со скоростью отклика простейших электронных компонентов. Как прямое следствие аппаратные реализации (при всей затратности их создания) устойчиво занимают нишу сверхбыстрых систем управления.

Заключительный вывод.
Сопоставляя промежуточные выводы №1 – №6, можно утверждать, что для весьма обширного класса технических систем программная реализация обладает значительными преимуществами по отношению к реализации аппаратной и может рассматриваться как основная платформа для построения таких систем управления.

IT-архитектура программной реализаций систем управления

В разделе «Сравнение аппаратной и программной реализаций систем управления» мы сосредоточили внимание на локальных системах управления или системах первого уровня. Заключительный вывод этого раздела дает нам возможность рассматривать взаимодействие систем первого и второго уровня с позиции программной реализации локальных систем. Первое, что необходимо отметить, это то, что микро компьютеры в составе локальных систем имеют сегодня достаточно развитый и универсальный интерфейс для организации внешнего взаимодействия. В рамках этого интерфейса, все рассмотренные нами микро компьютеры, предлагают **Ethernet**, **WiFi**,**USB.** Кроме того, хоть это было вынесено за скобки проведенных нами сравнений, эти микро компьютеры имеют встроенные графические процессоры или другие средства графического вывода через интерфесы **HDMI** и некоторые другие.

Локальная система как полнофункциональный компьютер

Наличие нескольких **USB -** портов и порта **HDMI** позволяют подключить к микро компьютеру клавиатуру, манипулятор «мышь» и дисплей, превратив локальный регулятор в полнофункциональную вычислительную среду на базе выбранной операционной системы (ОС). Такая вычислительная среда пригодна уже не только для эксплуатации, но также может использоваться для разработки, отладки и тестирования программ в условиях полного взаимодействии с реальным объектом. Кроме того, наличие четырех ядер в процессорах рассмотренных микро компьютеров при определенных настройках ОС позволяет организовать четыре параллельных потока выполнения программ в режиме, максимально приближенному к реальному времени.

Такие потоки можно соответственно распределить между:

- операционной системой и программами взаимодействия с внешней средой;
- программами, выполняющими алгоритм управления;
- программами контроля и оптимизации алгоритма управления;
- программами взаимодействия с пользователем системы управления, инструментам программирования, вспомогательным утилитам наблюдения и отладки, другими вспомогательными программами или (в режиме эксплуатации) распределить этот

поток к одному из трех названных выше. Как правило, этот поток присоединяется к программам, выполняющим алгоритм управления, что обеспечивает для них дополнительный запас по производительности.

Система управления второго уровня.

Несколько локальных систем управления в составе сложного, распределенного физического объекта, можно рассматривать как некоторый самостоятельный (абстрактный) объект управления. При этом основной задачей управления таким объектом определяется задача согласования работы локальных систем для обеспечения максимально эффективного функционирования распределенного физического объекта в целом. Решение такой задачи возлагается на систему управления второго уровня.

Сегодня реализация алгоритмов управления второго уровня все чаще рассматривается с позиции нейронных сетей. При этом возможны два подхода к сетевой архитектуре построения системы второго уровня:

- В качестве узлов нейронной сети рассматриваются непосредственно локальные управляющие системы;
- В качестве узлов нейронной сети рассматриваются узлы в программных моделях на отдельных вычислительных ресурсах (отдельных компьютерах).

Реализация нейронной сети с отдельными вычислительными ресурсами, сегодня можно считать более эффективным решением. Причиной тому создание фирмой Google в конце 2017 года системы искусственного интеллекта AlfaZero, а также фирмой Nvidia технология конкурирующих нейронных сетей (generative adversarial network, GAN). Основная идея AlfaZero и GAN заключалась в том, что две нейронные сети путем взаимодействия принуждали друг на друга эволюционировать. Если одна сеть эволюционировала в поиске спектра воздействий для нарушения функционирования другой сети, то другая эволюционировала в поиске моделей стабильного функционирования при таких комплексных (многомерных) воздействиях.

В нашем случае, если мы будем учитывать тенденции, намеченные ИИ AlfaZero, можно копию очередной **удачной** модели стабильного функционирования рассматривать как управляющую систему второго уровня. Главным фактором такого выбора является то обстоятельство, что отбраковка **неудачных** эволюционных вариантов осуществляется в независимой от физического объекта среде. Кроме того, сеть для поиска нарушающего стектра воздействий, благодаря возможности наблюдения за локальными подсистемами как на уровне их состояний, так и на уровне сигналов с АЦП/ЦАП, получает возможность конструировать такие спектры с реальных, а не абстрактных позиций.

Однако, какими бы адаптационными или эволюционирующими свойствами мы не наделяли алгоритмы системы управления второго уровня, они до получения доступа к управлению локальными системами, должны верифицироваться на виртуальных моделях системы, изолированных от управления физическим объектом. В противном случае аварии в виртуальном пространстве моделирования станут вполне осязаемыми физическими авариями.

Вывод №7. Очевидное требование верификации моделей второго уровня в условиях изоляции от управления физическим объектом, определяет преимущество реализации

систем второго уровня на отдельных вычислительных мощностях, что дополнительно освобождает ресурсы локальных систем для выполнения их непосредственных задач.

Для окончательного прояснения IT-архитектуры, необходимо рассмотреть механизм обмена информацией в обобщенной вычислительной среде, в которую мы разместим локальные системы управления и систему управления второго уровня.

Коммуникация между системами управления первого и второго уровней

В этом плане, обратим внимание на группу интерфейсов **Ethernet**, **WiFi**, на которых сегодня строятся компьютерные сети. Варианты компьютерных сетей на базе таких интерфейсов рассмотрены на рисунках 3 и 4.

Компьютерная сеть с преимущественно гальваническими каналами связи

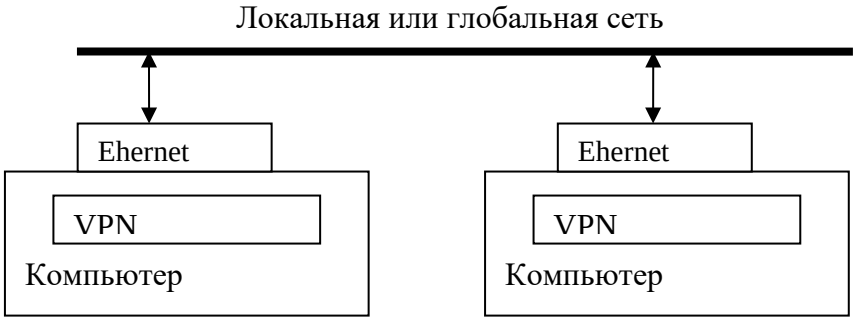

Рис 3. Локальная или глобальная компьютерная сеть

Если термин Ethernet достаточно хорошо знаком и, как правило, ассоциируется с устройством сетевая карта с локальными сетями на витых парах, то термин VPN (virtual personal network) расшифровывается как персональная виртуальная сеть.

> В операционных системах VPN часто реализуется в виде программных сервисов (Windows) или демонов (Unix), которые в ОС представимы как дополнительные, виртуальные сетевые карты. Однако с такими картами разрешаются осуществлять такие же операции, как и с реальными сетевыми картами. Основная задача VPN при передаче данных заключается в шифровании сетевых пакетов адресного уровня и размещение их специальных транспортных заголовках. Соответственно при приеме такие пакеты изымаются из транспортных заголовков и дешифрируются. Шифрование пакетов обеспечивает достаточно высокий уровень безопасности, поскольку любые пакеты, не соответствующие ключам шифрования, просто отбрасываются. Еще одной особенностью является возможность маршрутизации уже расшифрованных адресных пакетов. Эта особенность позволяет несколько сегментов локальной сети объединить в единую локальную сеть, безопасно пропустив трафик между сегментами через глобальный Internet. Следует также отметить, что в последнее время все более популярными становятся сетевые устройства с аппаратным выполнением функций VPN.

Компьютерная сеть с преимущественно радио каналами связи

Для ряда управляющих систем, в первую очередь для мобильных объектов, таких как беспилотные наземные, воздушные, морские аппараты, связь с системой управления второго уровня уже невозможно организовать через физические каналы с непосредственной гальванической связью (например, на витых парах различных категорий) или с оптической связью по оптоволоконным линиям.
В этом случае, как правило, приходится использовать передачу информации через радио каналы, а в отдельных случаях через акустические каналы (беспилотные морские аппараты). Однако, если наша система не имеет глобального характера (требующего применения спутниковой связи) или не управляет подводными аппаратами, то для таких мобильных объектов в технических системах используются интерфейсы WiFi.

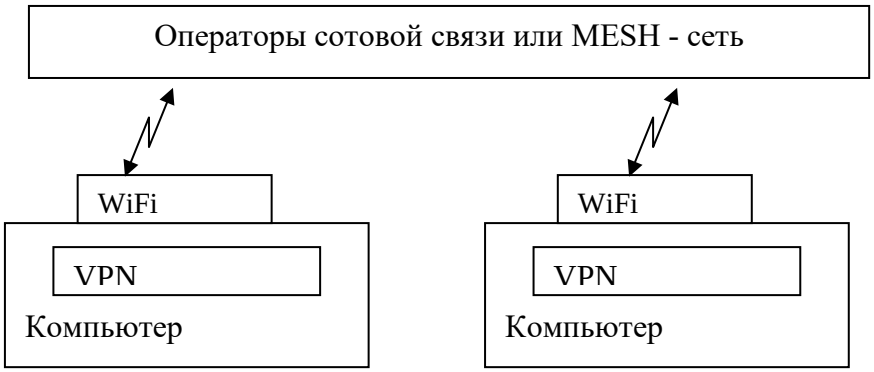

Рис 4. Сеть с соединением узлов посредством WiFi - радио канала.

Если интерфейс WiFi достаточно хорошо знаком, как популярное средство радио доступа к провайдеру через сотовый узел, то MESH – сеть требует дополнительных пояснений.

> MESH - сеть или ячеистая сетевая топология компьютерной сети, построенная на принципе ячеек, в которой рабочие станции сети соединяются друг с другом и способны принимать на себя роль промежуточных узлов связи для остальных участников. Как правило, узлы соединяются по принципу «каждый с каждым» что обеспечивает высокую отказоусточивость соединений. Однако, обычно для построения MESH – сети, используются специальные узлы маршрутизации, которые посредством WiFi радио каналов образуют основу такой сети, а рабочие станции (компьютеры) соединяются по WiFi только с такими узлами. По данным на 2017 год стоимость комплекта из пяти узлов маршрутизации составляла порядка US $230, а максимальное растояние устойчивой связи между узлами маршрутизации - порядка порядка 5 км.

Основным преимуществом MESH – сетей является их независимость от сот и провайдеров мобильного интернета, что позволяет развернуть MESH – сеть в полевых условиях и это особо важно для управления беспилотными аппаратами.
К качестве примера можно привести MESH - сеть Sonet (https://www.kickstarter.com/projects/sonnet/sonnet-decentralized-mobile-communication)

Очевидно, что для взаимодействия прикладного программного обеспечения, с помощью которого реализуются алгоритмы систем управления первого и второго уровней, необходимо также определить интерфейс подключения прикладных программ к сетевым средствам коммуникации. В соответствии со стандартами ISO/OSI (Open Systems

Interconnection basic reference model) эта модель определяет различные OSI - уровни сетевого взаимодействия для различных систем. Уровень OSI, на котором уже абстрагируются (скрываются) все специфические детали адресации узлов, особенности организации физических каналов и доставки сообщений между узлами, определяется как транспортный уровень OSI. На этом уровне узлы любой сети уже представлены как абстрактные адреса(IP-адреса), а конкретные программные подсистемы внутри узла представляются посредством номеров портов. Кроме того, сегодня все современные операционные системы обеспечивают прикладным программам доступ к транспортному уровню OSI с помощью наиболее популярных протоколов TCP (Transmission Control Protocol), UDP (User Datagram Protocol). Таким образом, протоколы TCP и UDP, а также наборы команд, с помощью которых осуществляется работа с этими протоколами, можно рассматривать как искомый нами интерфейс.

В то же время, следует подчеркнуть, что различным прикладным системам, может потребоваться информационное взаимодействие самыми различными способами (различными форматами построения структуры данных, передаваемых по сети, и/или различным порядком диалога узлов для обмена этими данными). Кроме того, если включать команды транспортных протоколов непосредственно в тексты прикладных программ, то алгоритмы таких программы станут сложно сопровождаемыми, поскольку прикладная логика программ будет переплетена с логикой оформления данных и организации сетевого взаимодействия узлов.

Клиент – серверная архитектура для управляющих систем

Традиционно названные проблемы устраняются путем включения между прикладными программами и сетевыми протоколами специальных организующих диалог и структуру данных подсистем. Такие подсистемы получили названия клиентских подсистем, серверных подсистем или комбинированных клиент – серверных подсистем, а соответствующие им узлы получили наименования серверов и клиентов. Иными словами, серверам определяется роль организации и обеспечения массового обслуживания, а множеству клиентов предоставляется возможность получать такое обслуживание. При этом прикладные программы (на серверах и клиентах) получают возможность вместо команд и форматов представления данных для протоколов TCP или UDP, использовать специально разработанные в подсистемах клиента и сервера, новые команды и форматы данных, ориентированные на прикладные смыслы и потребности систем управления.

Как известно в двухуровневой системе управления, локальные системы являются объектом управления для второго уровня. Кроме того, они разрабатываются, таким образом, чтоб в случае потери связи со вторым уровнем иметь возможность продолжать управление своей зоной ответственности в автономном режиме. Такие особенности локальных систем позволяют сформулировать основные роли первого и второго уровней управления, то есть:

- Локальные системы управления должны выступать в роли сервера, который предоставляет системе управления второго уровня сервисы для наблюдения и оптимизации режима своего функционирования. В этом случае система управления второго уровня выступает, по отношению к локальной системе управления, в роли клиента с полным набором прав по наблюдению и управлению.
- Локальны системы управления, в случае потери связи со вторым уровнем управления, могут принимать на себя роль клиента для получения от других локальных систем информации об их состоянии. Такая информация может

использоваться локальной системой для самостоятельного выбора или подстройки алгоритма управления в условиях возникшей изоляции.

- Система управления второго уровня взаимодействует не только с локальными системами управления, но и с техническим персоналом. В этом случае, она должна выступать в роли сервера, способного наделять клиентов (консоли технического персонала) индивидуальными наборами прав по наблюдению и управлению.
- Очевидно, что включение в коммуникацию консолей технического персонала, также накладывает на сервера локальных систем управления требование наделять клиентов (консоли технического персонала) индивидуальными наборами прав по наблюдению и управлению.

Обобщая клиент серверные роли, которые могут принимать на себя системы управления первого и второго уровней, можно предложить архитектуру платформы для реализации алгоритмов управления в типичных технических системах (рисунок 5).

Рис. 5. Клиент – серверная архитектура для управляющих систем

Основные итоговые выводы

1. Программная реализация распределенных систем управления имеет существенные преимущества по отношению к системам, которые реализованы исключительно аппаратными средствами даже, если это средства электроники. Исключение составляют только системы с требованием времени отклика менее чем одной - двух микросекунд.

2. При программной реализации модели управления первого (локальные системы управления) и второго уровней управления реализуются в виде программного кода, который выполняется под управлением операционных систем на современных компьютерах и микро компьютерах.
3. Для взаимодействия между системами управления первого и второго уровней используются клиент серверные подсистемы, которые принимают на себя все особенности взаимодействия со средой передачи данных и обеспечивая программы управления объектом логически совместимым с их функциями командным интерфейсом и форматами представления данных.
4. Унифицированный выход интерфейс на физическую среду передачи данных ISO/OSI, который обеспечивается во всех современных операционных системах (ОС), позволяет использовать для различных систем управления (далее узлов управляющей сети) различные операционные системы и технологии программирования. Такое гибкое применение ОС позволяет с одной стороны распределить разработку узлов между различными научно технологическими школами разработчиков, а с другой стороны, позволяет продлить жизненный цикл системы, применяя к узлам модульный принцип замены (обновления).
5. Возможность подключения множества параллельно работающих консолей технического персонала в различных точках среды передачи данных, при этом наделенных индивидуальным объемом прав по наблюдению и управлению, существенно повышает как процедуры разработки системы, так и процедуры эксплуатации.

Киев, 2018г.