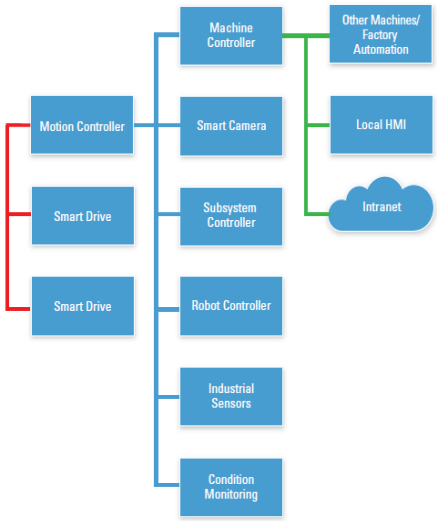
Эффективное использование производительности встраиваемых систем машинного зрения дает больше, чем просто зрение

Машинное зрение давно используется в системах автоматизации производства для улучшения качества продукции и производительности, заменяя традиционно выполняемый человеком визуальный контроль. Визуальные данные, получаемые в задачах захвата и перемещения, отслеживания объекта, метрологии, обнаружения дефектов и т.п., используются для улучшения производительности всей системы, предоставляя простую информацию "тест пройден/тест не пройден" или замыкая контуры управления. Использование машинного зрения не ограничивается автоматизацией производства; мы все являемся свидетелями массового внедрения камер в нашу повседневную жизнь: в компьютеры, мобильные устройства и особенно в автомобили. Автомобильные видеокамеры заднего вида появились всего несколько лет назад, а теперь автомобили поставляются с многочисленными камерами, предоставляющими водителю обзор в 360 градусов. Но, возможно, самый большой технологический прогресс в области машинного зрения связан с вычислительными мощностями. Поскольку производительность процессоров удваивается каждые два года, а их развитие фокусируется на технологиях параллельной обработки с использованием многоядерных процессоров (CPU), графических процессоров (GPU) и программируемых логических интегральных схем (FPGA), разработчики систем машинного зрения могут применять весьма сложные алгоритмы для визуализации данных и создания более интеллектуальных систем.

Усовершенствование технологии открывает новые возможности помимо просто более интеллектуальных или более мощных алгоритмов. Рассмотрим вариант дополнения промышленного станка средствами машинного зрения. Эти системы, показанные на рисунке 1, обычно разрабатываются как сеть интеллектуальных подсистем, образующих вместе распределенную систему, которая может быть спроектирована на основе модульного принципа. Однако по мере увеличения производительности системы использовать подход, основанный на аппаратных средствах, становится затруднительно, поскольку часто эти системы объединены смесью критичных и не критичных ко времени протоколов обмена данными. Соединение различных систем с использованием различных протоколов приводит к появлению узких мест, обусловленных задержками, потерей детерминизма и снижением пропускной способности. Например, если разработчик пытается создать приложение с подобной распределенной архитектурой и тесной интеграцией системы машинного зрения и системы управления движением, необходимой в визуальном сервоуправлении, он может столкнуться с проблемами производительности, ранее скрытыми недостатком возможностей обработки данных. Более того, поскольку у каждой подсистемы есть собственный контроллер, в действительности наблюдается даже *снижение* эффективности обработки, т.к. ни одна подсистема не нуждается в полной

вычислительной мощности всей системы. Наконец, из-за распределенной аппаратно-ориентированной архитектуры разработчикам приходится использовать несовместимые средства разработки для каждой подсистемы – специализированное программное обеспечение (ПО) для системы машинного зрения, специализированное ПО для системы управления движением и т.д. Это особенно сложно для небольших команд проектировщиков, где за многие компоненты проекта отвечает небольшая группа или даже один инженер.

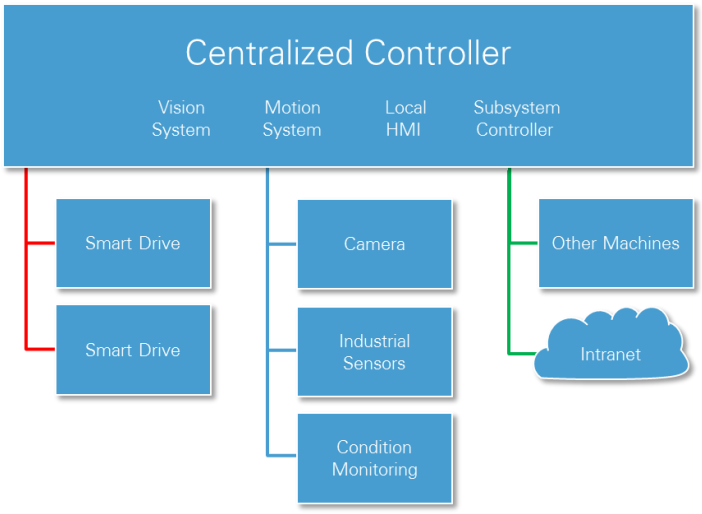


Motion Controller – контроллер управления движением; Smart Drive – интеллектуальный привод; Machine Controller – контроллер станка; Smart Camera – интеллектуальная камера; Subsystem Controller – контроллер подсистемы; Robot Controller – контроллер робота; Industrial Sensors – промышленные датчики; Condition Monitoring – мониторинг технического состояния; Other Machines/Factory Automation – другие механизмы/автоматизация производства; Local HMI – локальный интерфейс пользователя; Intranet – корпоративная сеть

Рисунок 1. Система из сети интеллектуальных подсистем, совместно образующих распределенную систему управления, реализуема по модульному принципу, но подобный аппаратно-ориентированной подход может стать причиной проблем достижения требуемой производительности.

К счастью, существует лучший способ разработки таких систем для современных станков и оборудования – способ, который уменьшает сложность, улучшает интеграцию, снижает риск и уменьшает время выхода на рынок. Что, если перестать концентрироваться на аппаратно-ориентированном видении задачи и сосредоточиться на программно-ориентированном подходе? Если использовать программные средства, предоставляющие возможность использования одного инструмента разработки для реализации различных задач, разработчики смогут отражать модульность механической системы в своем программном обеспечении.

Это позволит упростить структуру системы управления путем объединения различных задач автоматизации, в том числе визуального контроля, управления движением, ввода-вывода и человеко-машинного интерфейса в одной мощной встраиваемой системе. Это устраняет проблемы обмена данными между подсистемами, поскольку теперь все подсистемы работают в одном программном стеке на одном контроллере. Высокопроизводительная встраиваемая система машинного зрения - отличный кандидат на роль такого централизованного контроллера благодаря вычислительным мощностям, уже достигнутым в подобных устройствах.



Centralized Controller – централизованный контроллер: Vision System – система машинного зрения; Motion System – система управления движением; Local HMI – локальный интерфейс пользователя; Subsystem Controller – контроллер подсистемы.
Smart Drive – интеллектуальный привод; Camera – камера; Industrial Sensors – промышленные датчики; Condition Monitoring – мониторинг технического состояния; Other Machines – другие механизмы; Intranet – корпоративная сеть

Рисунок 2. Программно-ориентированный подход позволяет разработчикам упростить структуру системы управления путем объединения различных задач автоматизации, в том числе визуального контроля, управления движением, ввода-вывода и HMI в одной мощной встраиваемой системе.

Рассмотрим некоторые преимущества такой архитектуры централизованной обработки. Возьмем, например, приложение управления движением по видеоизображению, например, универсальный загрузчик, в котором система машинного зрения выдает указания системе управления движением. Здесь детали могут находиться в различных положениях и могут быть по-разному ориентированы. В начале задачи система машинного зрения получает изображение детали для определения ее положения и ориентации, и передает эту информацию системе управления движением. Затем система управления движением использует полученные координаты, чтобы переместить актюатор к детали и захватить ее. Она может также использовать эту информацию для корректировки ориентации детали перед ее размещением. При такой реализации разработчики могут исключить все приспособления, используемые ранее для ориентации и положения деталей. Это уменьшает стоимость и позволяет проще адаптировать приложение к конструкции новых деталей, с помощью лишь программной модификации.

Тем не менее, упомянутое выше ключевое преимущество аппаратно-ориентированной архитектуры – ее масштабируемость, которая главным образом объясняется связью между системами по Ethernet. Но необходимо уделить особое внимание и обмену данными по этой сети. Как было указано ранее, сложность такого подхода в том, что обмен данными по сети Ethernet недетерминирован, а пропускная способность ограничена. Для большинства задач управления

движением, где машинное зрение используется только в начале задачи, это приемлемо, но в других ситуациях изменение задержки может представлять проблему. Использование архитектуры централизованной обработки данных имеет несколько преимуществ. Во-первых, снижается сложность разработки, поскольку и систему управления движением, и систему машинного зрения можно разрабатывать в одной и той же среде программирования, и разработчику не нужно знать несколько языков или сред программирования. Во-вторых, потенциальное узкое место производительности системы в сети Ethernet устраняется, поскольку теперь данные передаются между циклами одного приложения, а не через физический уровень. В результате вся система работает детерминировано, поскольку она целиком выполняется в одном процессе. Это особенно важно, если машинное зрение необходимо использовать в цикла управления, например, в приложениях с визуальным сервоуправлением. Здесь система машинного зрения в процессе перемещения непрерывно получает изображения актюатора и целевой детали, пока перемещение не завершится. Эти получаемые изображения используются для обеспечения обратной связи об успешном выполнении движения. С помощью обратной связи разработчики могут улучшить точность существующей автоматики без необходимости приобретения высокопроизводительного оборудования для управления движением.

Теперь возникает вопрос: как выглядит такая система? Если разработчики собираются использовать систему, способную выполнять операции вычисления и управления, необходимые для системы машинного зрения, а также обеспечивающую бесшовную стыковку с другими системами – управления движением, HMI, вводом-выводом - они должны использовать аппаратную архитектуру, обеспечивающую требуемую производительность, а также интеллектуальные и управляющие способности, необходимые для каждой из этих систем. Хороший вариант для подобной системы – гетерогенная архитектура обработки данных, объединяющая процессор и FPGA с вводом-выводом. В архитектуру такого типа вложено много инвестиций, например, Xilinx Zynq All-Programmable SoC (системы на кристалле, объединяющие процессор ARM с Xilinx 7-Series FPGA), многомиллиардное приобретение компанией Intel компании Altera, множество систем машинного зрения, существующих на современном рынке, также используют эту архитектуру. Для систем машинного зрения использование FPGA предоставляет особые преимущества благодаря присущему FPGA параллелизму. Алгоритмы могут быть разделены так, чтобы выполняться тысячами разных совершенно независимых потоков. Но преимущества такой архитектуры касаются не только машинного зрения – она также придает множество преимуществ системам управления движением и ввода-вывода. Процессоры и FPGA могут использоваться для выполнения продвинутой обработки, вычислений и принятия решений. Разработчики могут подключиться практически к любому датчику по любой шине через каналы аналогового и цифрового ввода-вывода, промышленные и пользовательские протоколы, датчики, актюаторы, реле и т.д. Такая архитектура удовлетворяет и другим требованиям, например, требованиям таймирования и синхронизации, а также бизнес-задачам, например, продуктивности. Все хотят разрабатывать быстрее, а такая архитектура устраняет необходимость в больших специализированных командах разработчиков.

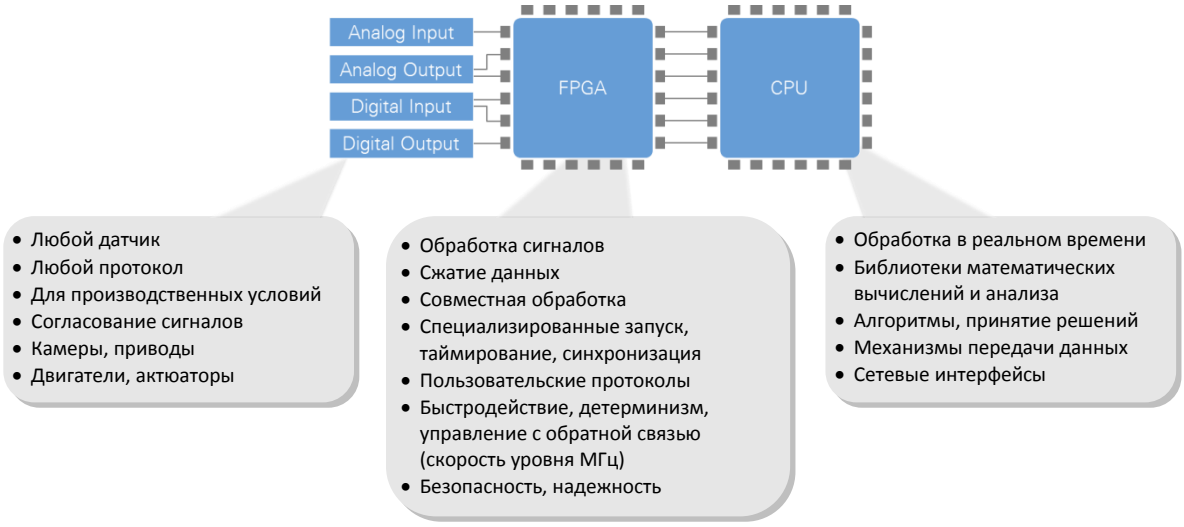


Рисунок 3. Гетерогенная архитектура, объединяющая процессор с FPGA и вводом-выводом – идеальное решение не только для разработчика высокопроизводительной системы машинного зрения, но и для интеграции управления движением, HMI и ввода-вывода.

К сожалению, хотя эта архитектура предлагает большие возможности для обеспечения производительности и масштабируемости, традиционный подход к ее реализации требует соответствующего опыта, особенно когда дело заходит об использовании FPGA. Это вносит значительный риск для разработчиков и может сделать использование архитектуры непрактичным или даже невозможным. Однако с помощью интегрированного программного обеспечения, например, NI LabVIEW, разработчики могут увеличить продуктивность и снизить риски, абстрагируясь от сложностей нижнего уровня и интегрируя все необходимые технологии в одной универсальной среде разработки, и это принципиальное отличие от любых других альтернатив.

Одно дело - обсуждать теорию, и совсем другое - увидеть применение теории на практике. Master Machinery – тайваньская компания, выпускающая установки для обработки полупроводников, одна из нах показана на рисунке 4. В этой конкретной установке используется сочетание машинного зрения, управления движением и промышленного ввода-вывода для извлечения кристаллов из кремниевой подложки и последующей их упаковки. Это отличный пример системы, где могла бы использоваться распределенная архитектура, подобная изображенной на рисунке 1 – каждая подсистема разрабатывается отдельно, а потом интегрируется в сеть. В среднем подобные станки обрабатывают примерно по 2000 элементов в час. Однако Master Machinery использовала другой подход. Они разработали свою установку на основе централизованной, программно-ориентированной архитектурой и встроили главный контроллер управления станком, системы машинного зрения и управления движением, ввод-вывод и HMI в один контроллер, программное обеспечение которого спроектировано в LabVIEW. Помимо снижения затрат, поскольку им не потребовалось разрабатывать отдельные подсистемы, такой подход значительно повысил производительность, поскольку теперь установка обрабатывает около 20000 элементов в час – в 10 раз больше, чем станки конкурентов.

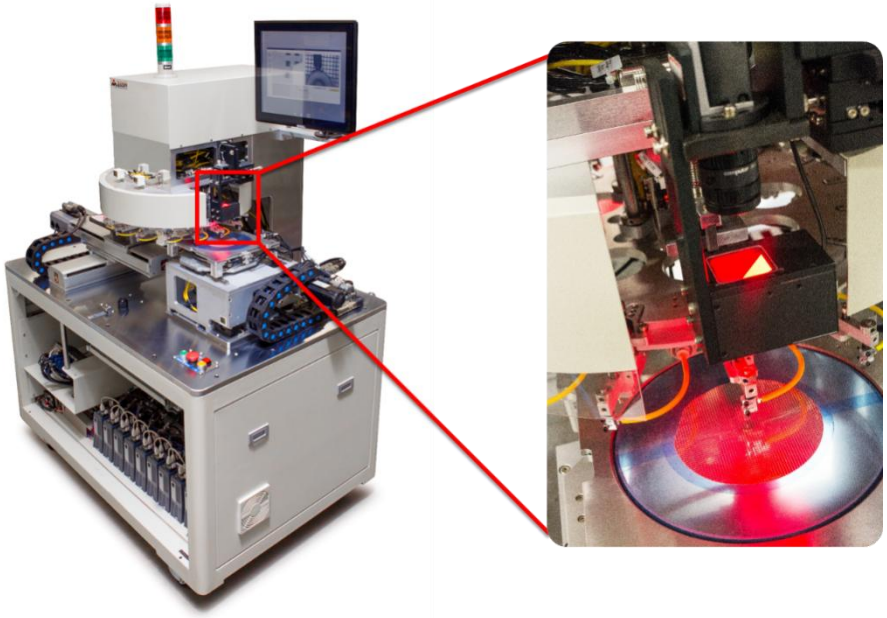


Рисунок 4. Используя централизованную, программно-ориентированную архитектуру, Master Machinery встроили главный контроллер управления станком, системы машинного зрения и управления движением, ввод-вывод и HMI в один контроллер, добившись повышения производительности в 10 раз по сравнению с конкурентами.

Ключевым компонентом в успехе Master Machinery стала возможность объединить несколько подсистем в один программный стек, в частности, системы машинного зрения и управления движением. Этот унифицированный подход позволил Master Machinery упростить не только проектирование системы машинного зрения, но и проектирование всей системы в целом.

Машинное зрение – сложная задача, требующая значительных вычислительных мощностей. Поскольку производительность обрабатывающих элементов – CPU, GPU, FPGA – растет по закону Мура, разработчики могут использовать эти компоненты для разработки весьма сложных алгоритмов. Разработчики могут также использовать эту технологию для улучшения производительности всех компонентов проекта, особенно в областях управления движением и ввода-вывода. При росте производительности всех этих подсистем используемая при проектировании традиционная распределенная архитектура оказывается перегруженной. Консолидация этих задач в одном контроллере с единым программным окружением устраняет узкие места из процесса разработки, и разработчики могут сосредоточиться на своих инновациях и не беспокоиться об их реализации.

Об авторе: Брендон Трис - старший менеджер по товарному маркетингу в области сбора данных и управления, сферой интересов которого является машинное зрение. Работает в NI, Остин, Техас.