ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ

"ТСА-Сервис"

ОКПД 2 26.51.70.190

УТВЕРЖДАЮ

Генеральный директор

ООО «ТСА-Сервис»

_____ Петров С.В.

«01» июня 2021 г.

Комплекс программно-технический Квинт-6

SCADA-система «Квинтегратор»

Расчётная станция

Руководство пользователя

ПФДИ.421457.009 И3.14

Москва

2021

Подп. и дата	
Инв. №	
Взаим. инв. №	
Подп. и дата	
Инв. № подл.	

Содержание

1 Введение

Настоящее руководство содержит правила работы с программным приложением **Расчётная станция**. Приложение входит в состав пакета **КВИНТегратор -** фирменного программного обеспечения программно-технического комплекса Квинт-6**.**

1.1 Назначение

При работе АСУТП производится сбор большого количества **первичных данных** (показаний датчиков, состояний механизмов и др.). Эти данные заносятся в Архивную станцию (или несколько Архивных станций) и отображаются на экранах Операторских станций или Станции анализа архивных данных.

Наряду с первичными данными, пользователям необходимы для наблюдения различные **вычисленные параметры**, являющиеся функциями от первичных данных.

Если функции расчета не очень сложны, их можно реализовать непосредственно в Ремиконтах например, с помощью алгоритмов ПКП и ВКП (Ремиконт – фирменное название программируемых контроллеров Квинта**)**.

Если вычисления сложны или требуют каких-либо недоступных Ремиконту ресурсов, следует использовать Расчётную станцию или Мезон.

В Квинте для сложных вычислительных задач или если эти задачи требуют каких-либо недоступных Ремиконту ресурсов **Вычислительная станция** может быть реализована либо с помощью программного приложения **Расчётная станция**, установленного на какой либо Рабочей станции, либо с помощью **Мезон-станции**, программируемой средствами **Мезон-редактора.**

Основным критерием выбора между **Расчётной станцией** и **Мезон-станцией** является квалификация составителя расчетов. Создание задач для Расчётной станции требует опыта программирования на языке **C++** или другом языке, для Мезона – опыта работы с диаграммами функциональных блоков.

Далее рассматривается варианта реализации **Вычислительной станции** средствами программного приложения **Расчётная станция** (далее по тексту – РС).

1.2 Архитектура

РС позволяет получать первичные данные в реальном времени из различных источников, вычислять расчетные параметры по составленному проектантом алгоритму и записывать результаты в различные приемники информации. Источниками и приемниками могут служить Ремиконты, Архивные станции Квинта и OPC-сервера, зарегистрированные в проекте АСУ ТП. Записанные данные доступны на операторских станциях и на станциях анализа также как и первичные данные.

Алгоритм расчета составляется проектантом на языке C++ или Pascal с использованием описанного ниже программного интерфейса **CsApi6**. Возможны 2 **варианта сборки** расчетов:

1 Алгоритм компилируется в виде **динамической библиотеки** (DLL), которая подключается к РС. В этом случае РС обеспечивает своевременный запуск расчета и контролирует его зависание.
2 Проектант создает **самостоятельное приложение** в виде EXE-файла. В этом случае проектант должен продумать организацию запуска и контроль выполнения этого приложения. Рекомендуется для этой цели использовать программную службу **Монитор приложений** из состава **КВИНТегратор**.

1.3 Инструментарий

1.3.1 Программирование

Для программирования расчётных задач необходима среда программирования или, как минимум, компилятор, запускаемый из командной строки. Необходимые заголовочные файлы и библиотеки входят в дистрибутивного пакета программ Квинта.

Разработчики Квинта создавали и тестировали расчётные задач, написанные в следующих средах:

- Microsoft Visual Studio 2005;
- Borland C++ Builder 6;
- Borland Delphi 6;

- Borland C++ compiler 5.5 (бесплатный).

1.3.2 Конфигурирование

Для подключения и настройки откомпилированных DLL к РС используется форма **Конфигурация расчетной станции,** которая входит в состав приложения **Аркада** пакета **САПР** Квинта**.**

1.3.3 Выполнение

Запуск РС должен производиться с помощью службы **Монитор приложений**. (Описание работы **Монитора приложений** и его настройки приведены в документе «Комплексы программно-технические Квинт-6. Монитор приложений. Руководство пользователя ПФДИ.421457.009 И3. 15»).

На этапе проектирования и наладки АСУ ТП допускается запуск РС из дерева пакетапрограмм **КВИНТегратор**.

Для полноценного функционирования РС необходимо наличие источников информации, например, Архивной станции Квинта. Архивную и Расчётную станцию можно запускать на одном и той же Рабочей станции или на разных Рабочих станциях.

Управление выполнением задач на РС производится с помощью программного приложения **Администратор серверов Квинта**. Описание работы **Администратора** приведены в документе «Комплексы программно-технические Квинт-6. Администратор серверов Квинта. Руководство пользователя ПФДИ.421457.009 И3. 16»). **Администратор** можно запускать на той же Рабочей станции, что и РС, или на любой другой из состава Квинт-6.

1.3.4 Отладка

Для отладки можно использовать:

- механизм отладочной печати (ее вывод направляется в Главное окно КВИНТегратора);
- приложение csdebug.exe, которое следует запускать из-под отладчика C++ (например Borland Turbo Debugger).

2 Этап проектирования

На этапе проектирования следует выполнить следующие действия:

- добавить в проект объекты-источники и объекты-приемники информации (если они еще не введены). Это можно сделать в форме **Объекты** приложения **Аркада**;
- написать и скомпилировать алгоритмы расчета (в подходящей среде программирования);
- сконфигурировать РС с помощью формы **Конфигурация расчетной станции** приложения **Аркада**. Этот этап не требуется, если алгоритмы компилируются в виде отдельного приложения.

2.1 Программирование задач

Алгоритмы расчётов составляются на языке C++ или Pascal в среде программирования или с помощью текстового редактора. Для получения чтения-записи данных и для взаимодействия с ядром РС расчетная задача должна использовать **программные интерфейсы (API)**.

Квинт-6 предоставляет 2 варианта **API**, условно называемые **старый** и **новый**.

Их функциональные различия приведены в таблице 1.

Таблица 1 – Функции и варианты поддерживающих их API

Функция	**Старый API**	**Новый API**
Позволяет получать текущие данные	Да	Да
Позволяет получать ретроспективные данные	Нет	Да
Периодичность выполнения задачи	Фиксируется в Аркаде	Фиксируется в Аркаде или определяется в коде задачи
Состав источников и приемников	Фиксируется в Аркаде	Определяется в коде задачи
Варианты сборки	Только DLL	DLL или EXE
Языки программирования	C++	C++ или Pascal

2.1.1 Старый программный интерфейс

2.1.1.1 Подключение

В исходный файл задачи следует включить заголовочный файл ***cs.h*** из папки ***[Квинт]\Include***. (Здесь и далее ***[Квинт]*** обозначает папку, в которую был установлен Квинт-6 из дистрибутивного пакета).

2.1.1.2 Основная процедура

В старом интерфейсе ядро РС периодически вызывает процедуру пользователя, передавая ей 2 массива:

1 Входные параметры.
2 Выходные параметры.

Процедура имеет следующий вид:

```
void __stdcall MyProc(const TValue* inputs, int inputCount, TWrite* outputs,
int outputCount)
```

Марки входных и выходных параметров указываются в приложении **Аркада** в форме **Конфигурация расчётной станции**. Там же указывается период вызова процедуры.

2.1.1.3 Пример

Пример задачи можно найти в файле ***[Квинт]\Samples\CS\cscopy.cpp***. В этом примере реализована функция ***CopyValues***, копирующая значения входных параметров в выходные. Скомпилированный модуль ***[Квинт]\Bin\CsCopy.dll*** можно использовать для целей копирования значений.

2.1.2 Новый программный интерфейс

2.1.2.1 Подключение

На языке **C++** в исходный файл задачи следует включить заголовочный файл ***[Квинт]\Include\CsApi6.h***. В проект нужно добавить библиотеку ***CsApi6.lib*** (для компиляторов Borland – из папки ***[Квинт]\Lib\Borland***, для компиляторов Microsoft – из папки ***[Квинт]\Lib\Microsoft***).

На языке Pascal в исходный файл задачи следует включить модуль ***[Квинт]\Include\CsApi6.pas***.

Примечание - Более ранняя версия этого интерфейса использовала файлы ***qCsApi.h***, ***qCsApi.lib*** и ***CsApi.pas***. Эти файлы включены в дистрибутив Квинта для совместимости и их можно использовать. Рекомендуется, по возможности, переходить на новый вариант, поскольку он:

- будет развиваться в дальнейшем;
- обеспечит совместимость на уровне скомпилированных модулей с Квинтом более ранних версий.

2.1.2.2 Основная процедура

В новом интерфейсе ядро РС также вызывает процедуру пользователя следующего вида:

На языке **C++:**

```
int __stdcall MyProc(const char* cmdLine)
```

На языке **Pascal**:

```
function MyProc(CmdLine: PChar): Integer; stdcall;
```

Новый интерфейс обеспечивает более гибкий подход по сравнению со старым, в частности:

- процедура сама запрашивает значения нужных параметров с помощью функций ***CSOpenParam*** и ***CSReadData***;
- процедура не обязана быстро завершаться, а может вместо этого входить в «спячку» посредством вызова ***CSSleep***.

При варианте сборки в виде DLL процедура пользователя обычно строится по следующей схеме:

```
int __stdcall MyProc(const char* cmdLine)
{
    ...                     // открываем параметры
    while(CSNeedWork())     // пока нет команды на завершение
    {
        ...                 // что-то считаем
        CSSleep(10. * 60); // ждем 10 минут
    }
    ...                     // закрываем параметры
}
```

Функция ***CSNeedWork*** возвращает ***false***, когда пользователь средствами приложения **Администратор серверов** выдаёт команду остановить задачу, или когда вся РС завершает работу.

Процедура пользователя должна либо достаточно быстро завершаться, либо регулярно вызывать функцию ***CSSleep***. Если она не вызывает ее в течение определенного времени, задача считается зависшей.

После завершения процедуры пользователя ядро закрывает все незакрытые параметры, принадлежащие данной задаче.

2.1.2.3 Инициализация

При использовании варианта сборки задачи в виде EXE-файла следует вначале, перед вызовом других функций CsApi, вызвать функцию ***CsInit***, а при завершении работы с CsApi – функцию ***CsDone***.

При сборке в виде DLL вызывать эти функции нет необходимости.

2.1.2.4 Обработка ошибок

Если при выполнении какой-либо функции CsApi возникает ошибка, функция возвращает оговоренное на этот случай значение и формирует описание ошибки. Описание ошибки можно получить с помощью функции Windows API ***GetErrorInfo*** или с помощью функции-обертки ***GetKvintError***. Пример на языке **Pascal**:

```
if not CsInit then
    raise Exception.CreateFmt('Ошибка инициализации: %s',  [GetKvintError]);
```

2.1.2.5 Параметры

Для работы с параметрамирасчетная задача должна вначале открыть параметр с помощью функций ***CSOpenParamByName*** или ***CSOpenParamById***.

Расчетная задача может получать текущее значение открытого параметра с помощью функции ***CSReadData*** и изменять значение (записывать в архив) с помощью функции ***CSWriteData***.

Если параметр больше не используется, можно закрыть его с помощью функции ***CSCloseHandle***. При завершении задачи все открытые параметры закрываются автоматически.

Пример на языке **C++**:

```
CSHandle param1 = CSOpenParamByName("Марка1", NULL, OP_READ);
if(param1 == NULL)
    return;
CSData data1;
if(CSReadData(param1, data1))
{
    double v = data1.Value;
//  здесь производим вычисления...
}
CSCloseHandle(param1);
```

Для упрощения работы с параметрами рекомендуется использовать inline-класс ***CSParam***.

Приведённый выше пример будет выглядеть следующим образом:

```
CSParam param1("Марка1", NULL, OP_READ);
if(!param1)
    return;
if(param1.Read())
{
    double v = param1;
//  здесь производим вычисления...
}
```

Функция ***CSCloseHandle*** вызывается неявно в деструкторе класса ***CSParam***.

2.1.2.6 Переменные

Переменные хранятся на самой РС, и только она может их использовать. Значения переменных автоматически сохраняются на жёстком диске РС и считываются при запуске задачи. Несколько задач могут использовать одну и ту же переменную.

Для работы с переменной расчетная задача должна вначале открыть ее с помощью функции ***CSOpenVariable***.

Для получения значения переменной следует использовать функцию ***CSReadData***, а для изменения - ***CSWriteData***

Если переменная больше не используется, можно закрыть ее с помощью функции ***CSCloseHandle***.

Пример на языке **C++:**

```
CSHandle var1 = CSOpenVariable("MyVar", 0);
if(var1 == NULL)
    return;
CSData data1;
if(CSReadData(var1, data1))
{
    double v = data1.Value;
//  здесь производим вычисления...
}
CSCloseHandle(var1);
```

Для упрощения работы с переменными рекомендуется использовать inline-класс ***CSVariable***. Приведённый выше пример будет выглядеть следующим образом:

```
CSVariable var1("MyVar", 0);
if(!var1)
    return;
if(var1.Read())
{
    double v = var1;
//  здесь производим вычисления...
}
```

2.1.2.7 Ретроспекция

Расчётная задача имеет доступ не только к текущим (последним) значениям параметров, но и к прошлым (ретроспективным) данным. Для этого используется функции ***CSFindFirst*** и ***CSFindNext*** . При открытии параметра в функции ***CSOpenParamByName*** следует указывать флаг OP_HISTORY (можно комбинировать его с флагами OP_READ и OP_WRITE с помощью оператора |).

Пример, где перебираются все значения параметра за последний час:

```
CSHandle param1 = CSOpenParamByName("Марка1", NULL, OP_HISTORY);
if(param1 == NULL)
    return;
QATime now = CSGetTime();                // получаем текущее время
CSData data1;
```

```
bool found = CSFindFirst(param1, data1, now - 60 * 60);
while(found && data1.Time < now)
{
    double v = data1.Value;
//  здесь производим вычисления...
    found = CSFindNext(param1,  data1);
}
CSCloseHandle(param1);
```

Тот же пример с использование класса CSParam:

```
CSParam param1("Марка1", NULL, OP_HISTORY);
if(!param1)
    return;
QATime now = CSGetTime();                // получаем текущее время
bool found = param1.FindFirst(now - 60 * 60);
while(found && param1.GetTime() < now)
{
    double v = param1;
//  здесь производим вычисления...
    found = param1.FindNext();
}
```

2.1.2.8 Пример

Пример использования CsApi6 можно найти в файле ***[Квинт]\Samples\CS\SampleNew.cpp***.

2.2 Конфигурирование РС

Конфигурирование РС осуществляется средствами приложения **Аркада (**см. документ «Комплексы программно-технические Квинт-6. Система управления технологической базой данных Аркада. Руководство пользователя ПФДИ.421457.009 И3. 4»).

В Аркаде необходимо выбрать пункт меню Конфигурация Расчётной станции.

Для добавления новой задачи нажмите кнопку "+", затем задайте свойства задачи.

Если задача имеет интерфейс старого типа, задайте для нее списки параметров.

Изменения в настройках РС вступят в силу только после запуска конфигуратора.

2.2.1 Задание свойств задачиЧтобы добавить новую задачу, необходимо нажать кнопку "+". Для этой новой задачи следует указать:

- **Имя задачи.** Это символьная строка, позволяющая в дальнейшем различать задачи;
- **Файл DLL**, в котором реализована задача. Если файл указан вместе с путем, то это должен быть локальный путь на компьютере расчётной станции;
- **Имя процедуры**. Его можно ввести вручную, либо выбрать из списка экспортируемых процедур данной DLL;
- **Период запуска** процедуры. Если период указать равным 0, процедура будет запущена однократно (по команде пользователя или по астрономическому времени). Если задан ненулевой период, и на момент очередного запуска задача еще выполняется, то очередной запуск просто пропускается;
- **Тайм-аут**, т.е. контрольный промежуток времени, в течение которого задача обязуется либо завершиться, либо (для нового интерфейса) вызывать функцию **`CSSleep`**. Если задача не сделает этого, она считается зависшей. Если указать тайм-аут равным 0, контроль зависаний производиться не будет;
- **Имя расчетной станции**, на которой будет запускаться данная задача. Станция выбирается из списка компьютеров, зарегистрированных в приложении **Администратор БД** и для которых в нём установлен флажок Расчётная станция.
- **Тип запуска** задачи:
    а) **Автоматически**. При этом задача запускается при старте РС;
    б) **Вручную**. При этом задача запускается пользователем средствами приложения Администратор серверов Квинта;
    в) **Отключена**. При этом задача не может быть запущена.

- **Тип интерфейса** задачи. Указание старого или нового интерфейса в соответствии с 2.1.

Дополнительно можно указать условия **запуска по абсолютному времени** и **командную строку** (передается в качестве аргумента в процедуру задачи в интерфейсе нового типа).

2.2.2 Задание списка параметров

Списки параметров нужно задавать только для задачи с интерфейсом старого типа.

В списке **Значения для чтения** формируется список исходных параметров, в числе которых могут быть как первичные параметры, получаемые архивной станцией от контроллеров, так и расчетные параметры (например, вычисленные другой расчетной станцией).

В списке **Значения для записи** формируется список выходных параметров. Все они должны относиться к оперативному типу **Параметр расчетный**.

2.3 Отладка задач

Квинт-6 не предоставляет собственного отладчика расчетных задач. Однако, можно использовать отладчики других фирм, либо применять отладочную печать.

2.3.1 Приложение CSDebug

Приложение ***CSDebug.exe*** представляет собой упрощенный вариант ядра РС, предназначенный для облегчения отладки задач. Отличия **CSDebug** от РС приведены в таблице 2.

Таблица 2 – Отладчики задач

РС	**CSDebug**
Запускает параллельно все зарегистрированные задачи	Запускает только одну задачу по выбору пользователя
Получает данные из архивной станции	Получает данные из эмулятора, что дает возможность изменять их вручную
Для контроля над выполнение задач требуется запускать Администратор серверов	Контроль над выполнением задачи производится непосредственно из приложения

Кроме того, CSDebug имеет режим "пошагового выполнения". В этом режиме задача автоматически переводится в режим **Пауза** при выходе из процедуры задачи, либо при вызове функции ***CSSleep***.

2.3.2 Использование отладчиков

В качестве отладчика можно использовать, например, **Borland C++ Builder** или **Borland Turbo Debugger**.

Для примера, рассмотрим **Borland C++ Builder**.

Последовательность действий такова:

- запустите **C++ Builder**;
- откройте проект расчетной задачи;
- вызовите пункт меню **Run/Parameters...** и в диалоговом окне укажите в качестве **Host Application** приложение **`cs.exe`** (собственно РС) или **`csdebug.exe`** из подкаталога **Bin** Квинта;
- в окне с исходным текстом задачи установите точку останова (**Breakpoint**) на процедуру задачи;
- запустите процесс (меню **Run/Run**). При запуске задачи управление должно передаться отладчику.

2.3.3 Отладочная печать

Под отладочной печатью понимается вставка в исходный текст программы операторов вывода текстовых сообщений, которые в ходе выполнения будут попадать в Главное окно консоли КВИНТегратора или **Монитора приложений** (в зависимости от того, откуда была запущена РС).

В интерфейсе нового типа для отладочной печати следует использовать функцию ***CSPrint*** или ***CSPrintV***. Их синтаксис аналогичен стандартным функциям ***printf*** и ***vprintf***.

Пример:

```
CSParam param1("Марка1", "Значение сигнала", OP_READ);
if(!param1)
    return;
if(param1.Read())
{
    double v = param1;
    CSPrint("Получили значение: Марка1 = %f\n", v);
}
```

Вывод текстов с помощью ***CSPrint*** и ***CSPrintV*** можно включить или отключить в приложении **Настройки отладочной печати** с помощью флажка **CS_User**.

2.3.4 Сообщения об ошибках

Если в процессе работы расчетная задача обнаружила какую-либо ошибку, она может передать текст ошибки с помощью функции ***CSSetErrorText***. Текст ошибки по каждой задаче можно увидеть в приложении **Администратор серверов Квинта**.

3 Этап выполнения

3.1 Запуск РС

Запустить РС можно следующими способами:

- штатно средствами службы **Монитор приложений** (рекомендуется). Для этого в настройках Монитора добавьте **cs.exe** в список приложений;
- при разработке проекта и отладке АСУ ТП из **КВИНТегратора**. Для этого необходимо выполнить двойной щелчок по узлу **Расчетная станция** в папке **Выполнение** в дереве приложений КВИНТегратор.

3.2 Контроль выполнения задач

Выполнение задач на РС можно контролировать с помощью приложения **Администратор серверов Квинта**, причем его можно запускать как на том же компьютере, что и РС, так и на любом другом компьютере.

В **Администраторе серверов** выберите узел **Расчетные станции**, нажмите кнопку **Добавить сервер** и укажите имя компьютера РС (если его нет в дереве).

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопро-водительного докум. и дата	Подпись	Дата
	изменен-ных	заменен-ных	новых	аннули-рованных					

Подп. и дата | Инв. № | Взаим. инв. № | Подп. и дата | Инв. № подп.