# Схема работа с дравером JPOS для проверки кодов маркировки.

Для проверки кодов маркировки в ФР есть следующие команды:

0xFF61, Проверка маркированного товара  
0xFF69, Принять или отвергнуть введенный код маркировки  
0xFF67, Привязка маркированного товара к позиции

Приложение сначала проверяет все коды маркировки, после чего регистрирует товары и привязывает к ним коды маркировки, затем закрывает чек. Последовательность команд для чека с двумя товарами:

* Открыть чек
* Проверка маркированного товара 1
* Принять код маркировки товара 1
* Проверка маркированного товара 2
* Принять код маркировки товара 2
* Регистрация товара 1
* Привязка товара 1 к позиции
* Регистрация товара 2
* Привязка товара 2 к позиции
* Закрытие чека

## Команды ФР

### Проверка маркированного товара

Код команды FF61h. Длина сообщения: 10+X+Y байт.

Пароль оператора (4 байта)

Входные данные (4+X+Y байт)

Состав входных данных:

| **Смещение** | **Длина** | **Параметр** | **Значение** |
| --- | --- | --- | --- |
| 0 | 1 байт | Планируемый статус | Тег 2003 |
| 1 | 1 байт | Режим обработки | Тег 2102 (сейчас всегда «0») |
| 2 | 1 байт | Длина кода маркировки (КМ) в байтах (X) | Полная длина КМ |
| 3 | 1 байт | Длина списка TLV в байтах | Полная длина списка TLV |
| 4 | X байт | КМ | Сам КМ, как он был прочитан сканером |
| 4+X | Y байт | Список TLV | Если планируется частичное выбытие маркированного товара (согласно с тегом 2003), то необходимо сформировать буфер из тегов 2108 (мера) и 1023 (количество) и передать его здесь |

Ответ: FF61h. Длина сообщения: 9+X байт.

Код ошибки (1 байт)

Результат проверки (6+N байт)

Состав данных результата проверки:

| **Смещение** | **Длина** | **Параметр** | **Значение** | **Примечание** |
| --- | --- | --- | --- | --- |
| 0 | 1 байт | Статус локальной проверки | Тег 2004 |  |
| 1 | 1 байт | Причина, по которой не была проведена локальная проверка | См. Примечание 1 ниже |  |
| 2 | 1 байт | Распознанный тип КМ | Тег 2100 |  |
| 3 | 1 байт | Длина дополнительных параметров | Длина данных, идущих далее | Если автономный режим, то «0» |
| 4 | 1 байт | Код ответа ФН на команду онлайн-проверки | В соответствии и вводом ошибки ФН | Если 0x20, то в следующем байте возвращается причина в соответствии с Примечанием 2 ниже  Значение 0xFF, если сервер не ответил в течение таймаута. |
| 5 | 1 байт | Результат проверки КМ | Тег 2106 | Только если сервер ответил без ошибок |
| 6 | N байт | Список реквизитов ответа сервера | TLV List | Только если сервер ответил без ошибок |

**Примечание:**

1 – Причина того, что КМ не проверен в ФН:

0 – КМ проверен в ФН

1 – КМ данного типа не подлежит проверки в ФН

2 – ФН не содержит ключ проверки кода проверки этого КМ

3 – Проверка невозможна, так как отсутствуют идентификаторы применения GS1 91 и / или 92 или их формат неверный.

4 – Проверка КМ в ФН невозможна по иной причине

2 – Причина проблемы при обработке ответа:

1 – Неверный фискальный признак ответа;

2 – Неверный формат реквизиов ответа;

3 – Неверный номер запроса в ответе;

4 – Неверный номер ФН;

5 – Неверный CRC блока данных;

7 – Неверная длина ответа.

### Принять или отвергнуть введенный код маркировки

Код команды FF69h. Длина сообщения: 7 байт.

Пароль оператора (4 байта)

Решение (1 байт):

0 – Отвергнуть

1 – Принять

Ответ: FF69h. Длина сообщения: 4 байта.

Код ошибки (1 байт)

Результат проверки (1 байт) тег 2106

**Примечание:**

Команду необходимо подавать после проверки каждого КМ!

### Привязка маркированного товара к позиции

Код команды FF67h. Длина сообщения: 7+N байт.

Пароль оператора (4 байта)

Длина кода маркировки (1 байт)

Данные маркировки (N байт)

Ответ: FF67h. Длина сообщения: 6+(6+N)1 байт.

Код ошибки (1 байт)

Первые 2 байта значения реквизита «код товара» (2 байта)

Тип Data Matrix (1 байт)

0 – КМ 88

1 – КМ симметричный

2 – КМ Табачный

3 – КМ 44.

Результат проверки (6+N байт)1 может отсутствовать (см. Примечание 1)

**Примечание:**

1 – В случае если код маркировки ранее не проверялся ранее командой FF61h «Проверка маркированного товара», ККТ сама её подаст и добывит ответ на неё к ответу на команду FF67h.

### Запрос статуса по работе с кодами маркировки

Код команды FF70h. Длина сообщения: 6 байт.

Пароль оператора (4 байта)

Ответ: FF70h. Длина сообщения: 11 байт.

Код ошибки (1 байт)

Статус работы с кодами маркировки (8 байт)

| **Наименование** | **Тип** | **Длина** | **Описание** |
| --- | --- | --- | --- |
| Состояние по проверке КМ | Byte | 1 | 0 – Таблица проверки КМ переполнена  1 – Нет КМ на проверке  2 – Передан КМ в команде B1h  3 – Сформирован запрос о коде маркировки при помощи команды B5h  4 – Получен и передан в ФН ответ на запрос при помощи команды B6h |
| Состояние по формированию уведомления о реализации маркированного товара | Byte | 1 | 0 – Уведомление о реализации маркированного товара не формируется  1 – Начато формирование уведомления о реализации маркированного товара  2 – Формирование уведомлений заблокировано из-за переполнения области временного хранения |
| Флаги разрешения команд работы с КМ | Byte | 1 | См. таблицу «[Флаги разрешения команд работы с КМ](#Флаги_разрешения_команд_работы_с_КМ)» ниже |
| Количество сохранённых результатов проверки КМ | Byte | 1 | Количество КМ, результаты проверки которых сохранены в ФН командой B2h c кодом «1» |
| Количество КМ, включенных в уведомление о реализации маркированного товара | Byte | 1 |  |
| Предупреждение о заполнении области хранения уведомлений о реализации маркированного товара | Byte | 1 | В этом параметре ФН информирует ККТ о заполнении области хранения уведомлений о реализации маркированного товара. Возможные следующие значения параметра:  0 – Область заполнена менее чем на 50%  1 – Область заполнена от 50 до 80%  2 – Область заполнена от 80 до 90%  3 – Область заполнена более чем на 90%  4 – Область полностью заполнена, формирование новых уведомлений невозможно |
| Количество уведомлений в очереди | Uint16, LE | 2 | Количество неподтверждённых или невыгруженных уведомлений о реализации маркированного товара |

Методы драйвера для вызова команд проверки кодов маркировки:

public static final int SMFPTR\_DIO\_CHECK\_ITEM\_CODE = 0x62;

public static final int SMFPTR\_DIO\_ACCEPT\_ITEM\_CODE = 0x63;

public static final int SMFPTR\_DIO\_BIND\_ITEM\_CODE = 0x64;

public static final int SMFPTR\_DIO\_CHECK\_ITEM\_CODE2 = 0x7F;

public static final int SMFPTR\_DIO\_FS\_READ\_MC\_STATUS = 0x85;

// 0xFF61, check item marking code

int itemStatus **=** 0**;**

int checkMode **=** 0**;**

byte**[]** mcData **=** **null;**

byte**[]** tlvData **=** **null;**

Object**[]** params **=** **new** Object**[**10**];**

params**[**0**]** **=** **new** Integer**(**itemStatus**);**

params**[**1**]** **=** **new** Integer**(**checkMode**);**

params**[**2**]** **=** mcData**;**

params**[**3**]** **=** tlvData**;**

printer**.**directIO**(**SmFptrConst**.**SMFPTR\_DIO\_CHECK\_ITEM\_CODE2**,** **null,** params**);**

int localCheckStatus **=** **(**Integer**)**params**[**4**];**

int localErrorCode **=** **(**Integer**)**params**[**5**];**

int symbolicType **=** **(**Integer**)**params**[**6**];**

int serverErrorCode **=** **(**Integer**)**params**[**7**];**

int serverCheckStatus **=** **(**Integer**)**params**[**8**];**

byte**[]** serverTLVData **=** **(**byte**[])** params**[**9**];**

// 0xFF69, accept item code

int action **=** 1**;**

params**[**0**]** **=** **new** Integer**(**action**);**

printer**.**directIO**(**SmFptrConst**.**SMFPTR\_DIO\_ACCEPT\_ITEM\_CODE**,** **null,** params**);**

int errorCode **=** **(**Integer**)**params**[**1**];**

// 0xFF67, bind item code

byte**[]** data **=** **null;**

params**[**0**]** **=** data**;**

printer**.**directIO**(**SmFptrConst**.**SMFPTR\_DIO\_BIND\_ITEM\_CODE**,** **null,** params**);**

int itemCode **=** **(**Integer**)**params**[**0**];**

int codeType **=** **(**Integer**)**params**[**1**];**

localCheckStatus **=** **(**Integer**)**params**[**2**];**

localErrorCode **=** **(**Integer**)**params**[**3**];**

symbolicType **=** **(**Integer**)**params**[**4**];**

serverErrorCode **=** **(**Integer**)**params**[**5**];**

serverCheckStatus **=** **(**Integer**)**params**[**6**];**

serverTLVData **=** **(**byte**[])** params**[**7**];**

// 0xFF70, FS read MC status

Object**[]** params **=** **new** Object**[**7**];**

printer**.**directIO**(**SmFptrConst**.**SMFPTR\_DIO\_FS\_READ\_MC\_STATUS**,** **null,** params**);**

int status **=** **(**Integer**)** params**[**0**];**

int notificationStatus **=** **(**Integer**)** params**[**1**];**

int commandFlags **=** **(**Integer**)** params**[**2**];**

int resultSavedCount **=** **(**Integer**)** params**[**3**];**

int realizationCount **=** **(**Integer**)** params**[**4**];**

int storageSize **=** **(**Integer**)** params**[**5**];**

int messageCount **=** **(**Integer**)** params**[**6**];**

# Схема без ожидания ответа на проверку кода маркировки

Больших задержек получения ответа не может быть, потому что в ФР настраивается таймаут ответа. Значение таймаута по умолчанию: 5 секунд. Если постоянно таймаут выходит, то приложение может не выполнять проверку кода маркировки и не привязывать его к товару.