

**EMM Enhancements  
on Fendt Machines**

**AGCO CONFIDENTIAL**

Revision: 1.0

Date: July 27th 2016

# Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comment** |
| 1.0 | 2016-07-27 | Nothdurft, Tobias | Initial Version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Definitions and Abbreviations

**ECU** – Electronic Control Unit

Content

[Document History 2](#_Toc457403414)

[Definitions and Abbreviations 3](#_Toc457403415)

[1 Objective 5](#_Toc457403416)

[2 General Approach 5](#_Toc457403417)

[3 EMM Master Requirements 5](#_Toc457403418)

[4 Interface Description 5](#_Toc457403419)

[5 ACM Requirements 6](#_Toc457403420)

[6 Detailed Interface Description 8](#_Toc457403421)

# Objective

The current EMM implementation on Fendt machines does not support a proper integration of the AGCO Connectivity Module (ACM). There are two shortcomings today:

1. It is not possible for the ACM to detect when an error goes inactive
2. Due to a boot time of up to 10 seconds, it can’t be guaranteed, that the ACM “sees” all the error notifications that the EMM master sends to the terminal and the dashboard.

The seconds of those issues shall be addressed by this document.

# General Approach

In order to cope with the boot time of the ACM, the EMM Master needs to buffer the received errors.

In order to keep interfaces consistent, the ACM shall use a similar interface and mechanism for requesting the occurred errors as the terminal.

# EMM Master Requirements

The EMM Master shall use the same internal mechanism for buffering error codes as it already uses for buffering errors to be shown on the terminal.

Within this circular buffer the EMM Master shall store all errors that shall be shown on the terminal and on the dashboard. All of those buffered errors shall be then send to the ACM on request.

**TODO: Buffer size?**

# Interface Description

The following GD index shall be used by the ACM to query the occurred errors from the EMM Master:

GD\_ID\_EMM\_ACM\_TRANSFER\_R = 15939, //0x3E43

See Section 6 for a detailed description of this GD index.

# ACM Requirements

Figure 1 shows how the information about occurred errors gets from an EMM Slave to the ACM as a sequence diagram.

The ACM has to resend the request after a cold start and a warm start of the EXT. This can be detected, if the EMM Master logged on again on the GD bus.

In the example shown in Figure 1, there is no active error stored in the EMM Master before the ACM requests the errors for the first time. Therefore, the EMM Master reports “0xFF = no error” back to the ACM. Shortly after that, an EMM Slave reports an active error to the EMM Master and the EMM Master forwards this information to the ACM.



Figure 1: Sequence Diagram of the Error Communication to the ACM– Initial Communication and Error is Reported as Active Afterwards

In the example shown in Figure 2, an EMM Slave reports that an error, which was reported by the EMM Slave as active before is not active any longer. Also this information is passed on to the ACM.



Figure 2: Sequence Diagram of the Error Communication to the ACM – an Error is Reported as Inactive

# Detailed Interface Description

GD\_ID\_EMM\_ACM\_TRANSFER\_R = 15939, //0x3E43  
///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////  
// Function: Transfer error to the AGCO Connectivity Module (ACM) with hand shake  
// Access: r  
// Signals:  
// ---------------------------------------------------------------------------------------------------------------------------------------  
// Signal Name                             |Byte|Bit|Length|DataType|Factor|Offset|   Unit   |  Min |  Max  
// ---------------------------------------------------------------------------------------------------------------------------------------  
// Error\_code                              |  6 | 0 |   8  |  UINT  |     1|     0|          |     0|   255  
// Error\_source                            |  7 | 0 |   8  |  UINT  |     1|     0|          |     0|   255  
//  
///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////