High Level Description

PROTOCOL STACK INTERFACE (PSI)

Document Number:	8462.720.01.001
Version:	1.1
Status:	Updated
Approval Authority:	
Creation Date:	2004-Jan-12
Last changed:	2015-Mar-08 by Ricarda Marzillier
File Name:	HLD_PSI.doc

Important Notice

Texas Instruments Incorporated and/or its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products, software and services at any time and to discontinue any product, software or service without notice. Customers should obtain the latest relevant information during product design and before placing orders and should verify that such information is current and complete.

All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment. TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI products, software and/or services. To minimize the risks associated with customer products and applications, customers should provide adequate design, testing and operating safeguards.

Any access to and/or use of TI software described in this document is subject to Customers entering into formal license agreements and payment of associated license fees. TI software may solely be used and/or copied subject to and strictly in accordance with all the terms of such license agreements.

Customer acknowledges and agrees that TI products and/or software may be based on or implement industry recognized standards and that certain third parties may claim intellectual property rights therein. The supply of products and/or the licensing of software does not convey a license from TI to any third party intellectual property rights and TI expressly disclaims liability for infringement of third party intellectual property rights.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products, software or services are used.

Information published by TI regarding third–party products, software or services does not constitute a license from TI to use such products, software or services or a warranty, endorsement thereof or statement regarding their availability. Use of such information, products, software or services may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Change History

Date	Changed by	Approved by	Version	Status	Notes
2004-Jan-12	Ricarda Marzillier		0.1	Draft	1
2004-Feb-25	Ricarda Marzillier		1.0	Approved	2
2004-Sep-09	Ricarda Marzillier		1.1	Update	3
2005-Jan-18	Ricarda Marzillier		1.1	Update	4

Notes:

1. Initial version
2. Corrections after review; using PTG template
3. Update: remove ATP/AAA

4. Update history, version number, status

Table of Contents

List of Figures and Tables

List of References

[7010.801]	References and Vocabulary, Texas Instruments
[10-08-03-FIO-CS-65-68-UPSI]	Feature Implementation Options F-CS-65 and F-CS-68

1 Introduction

Today the Protocol Stack provides several interfaces to internal and external applications. Each interface needs a different configuration, uses a different protocol, and needs to be tested separately. But all applications use AT commands to configure the Protocol Stack and after that they exchange data with the Protocol Stack.
As described in [2] a unified interface for the Protocol Stack (PSI - Protocol Stack Interface) is proposed which increases the flexibility, reduces the effort of maintenance, and simplifies the internal handling and the tests. In addition it simplifies extensions to new devices and new internal applications because only a small layer needs to be adapted.

This document describes the current implementation status for commands, serial and packet data exchange with respect to certain hardware configurations (Calypso, Perseus and Helen) and the implementation phases needed to introduce the new entity PSI. Because of customer requirements two short-term (phase 1a + 1b) and a long-term (phase 2) solutions are necessary. In phase 1a the entities UART and PKTIO are completed to DIO interface, version 4. Additional the multiplexer part of UART is shifted to separate driver software. Phase 1b covers the access to USB driver via the new entity PSI. Finally in phase 2 the entities UART and PKTIO are replaced by the entity PSI.

2 Current implementation status for data exchange and AT commands

2.1 Perseus1/Perseus2

This hardware configuration bases on a two-processor solution. The High Level Operating System is connected with our protocol stack via shared memory.

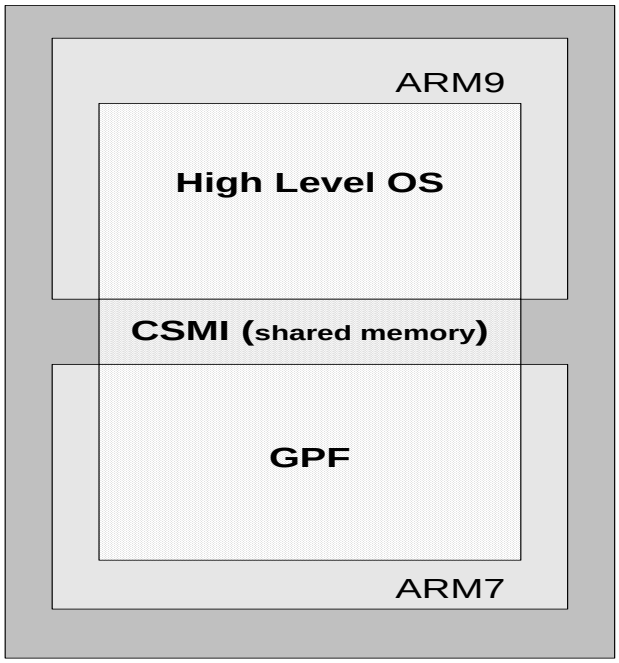

Figure 1: Simplified plan of Perseus1/Perseus2 (Avenger 2 resp. P2- or F-Sample)

2.1.1 Serial Data

The UART entity sends and receives serial data via UF part of the UART driver. The data multiplexing is not needed because the shared memory layer CSMI provides multiple data channels. The DTI Control/Connect Manager of ACI starts the data flow by connecting of UART entity with the requested target entity (CSD UP resp. PPP Server).

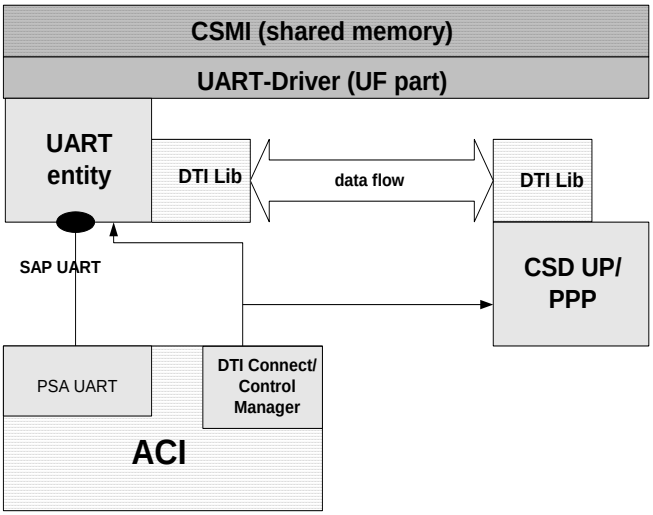

Figure 2: Serial data and command flow at Perseus1/Perseus2

2.1.2 Packet Data

Currently the DIO driver is interface between the shared memory layer CSMI and the Protocol Stack. DIO, version 3, provides available devices with their capabilities to entity PKTIO. PKTIO informs the ACI which can control several devices for packet data. Via AT commands (AT%DINF, AT%DATA) the user can configure the data channels. The DTI Control/Connect Manager of ACI initiates the data flow by connecting of DIO with the requested target entity SNDCP (PS UP). The shared memory layer CSMI offers some channels for packet data. In this case data multiplexing is not necessary.

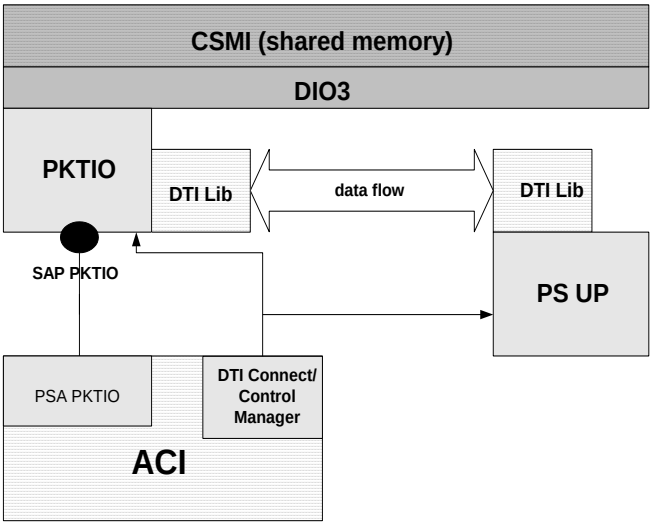

Figure 3: Packet data at Perseus1/Perseus2

2.2 Helen1 / Helen2

This hardware configuration bases on a two-processor solution. The High Level Operating System is connected with our protocol stack via serial line.

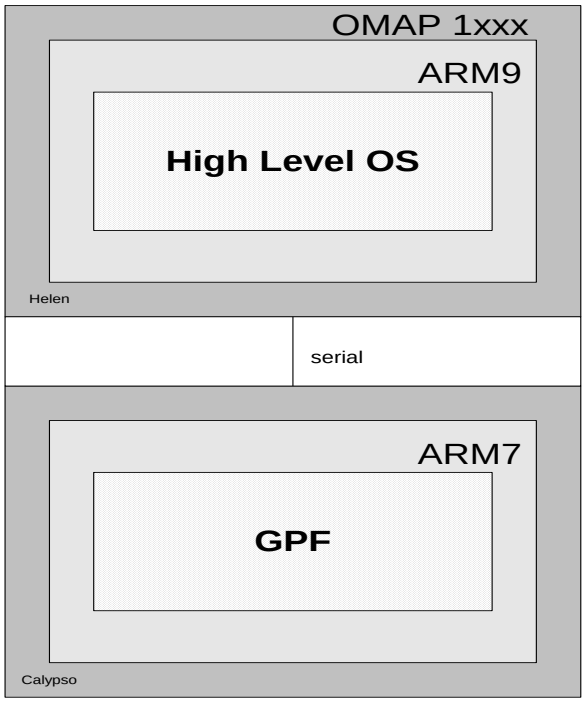

Figure 4: Simplified plan of Helen1/ Helen2 (PDA resp. H-Sample)

2.2.1 Serial Data

The UART entity sends and receives serial data via UF part of the UART driver. If several UART connections are requested the multiplexing part of driver is used because there is only one serial connection to the application part of Helen. The DTI Control/Connect Manager of ACI starts the data flow by connecting of UART entity with the requested target entity (CSD UP resp. PPP Server).

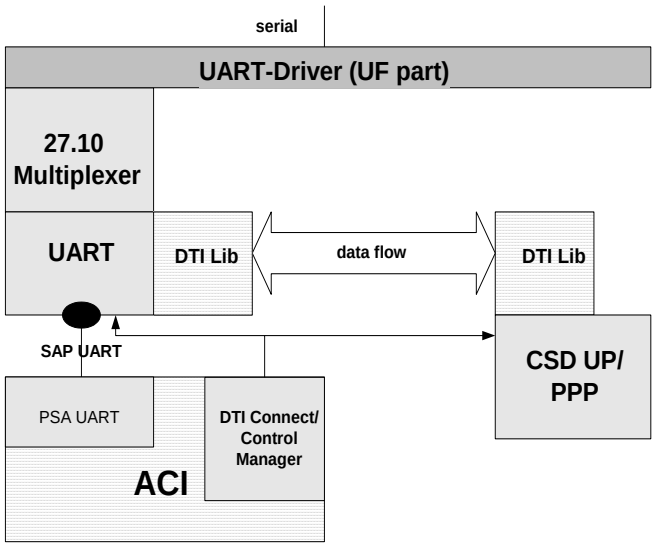

Figure 5: Serial data and command flow at Helen1/Helen2

2.2.2 Packet Data

Currently a solution for exchanging of packet data at this hardware configuration is not implemented.

3 Protocol Stack Interface

Up to now the entities PKTIO and UART convert the functional interface from the several application layer to primitives. The new entity PSI replaces all of these protocol stack entities. The interface layer DIO, version 4, provides the unified functional interface between each application layer and PSI.

Advantage:

- Easier enhancements of protocol stack access for further driver (f.e. USB) that means they have to support the DIO interface, vs.4
- Only maintenance for one entity (PSI) which uses DIO interface, vs.4 to each kind of application layer
- Reduced test effort
- Reduced SAP's for connection with ACI

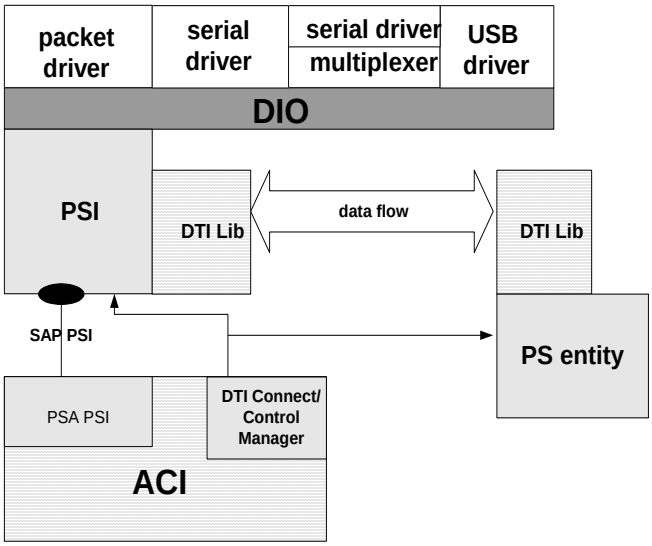

Figure 6: data and command flow for each kind of data within phase 2

3.1 Implementation Phase 1A

3.1.1 Perseus1/Perseus2

3.1.1.1 Serial Data

The UART entity has to include the adaptation for DIO interface via functional calls. For the UART driver is needed a glue layer to DIO interface too. The access to ACI is unchanged.

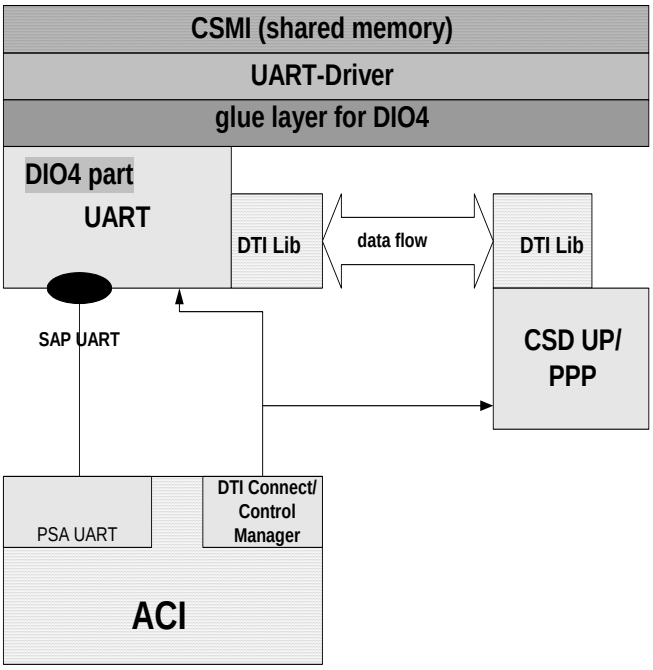

Figure 7: Serial data and command flow within phase 1a at Perseus1/Perseus2

3.1.1.2 Packet Data

The entity PKTIO needs adaptations to DIO vs. 4. The access to ACI is unchanged.

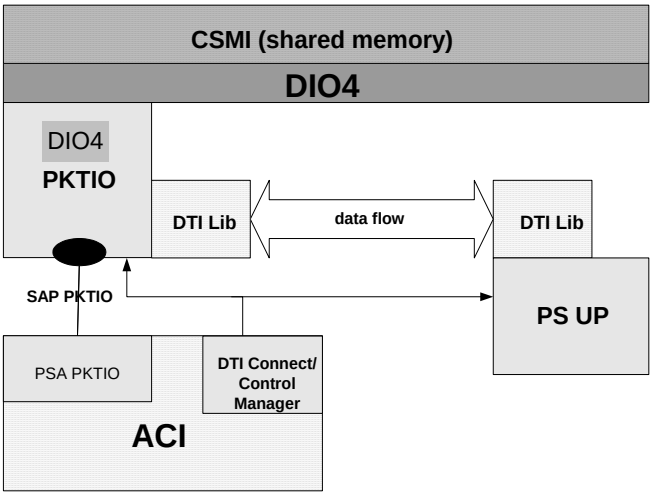

Figure 8: Packet data and command flow within phase 1a at Perseus1/Perseus2

3.1.2 Helen1/ Helen2

3.1.2.1 Serial Data

The multiplexer part of the UART entity has to been shifted to separate driver software with access to DIO interface, vs. 4. The UART entity must support DIO interface via a functional interface. For the UART driver is needed an adaptation (glue layer) to the DIO driver. The access to ACI is unchanged.

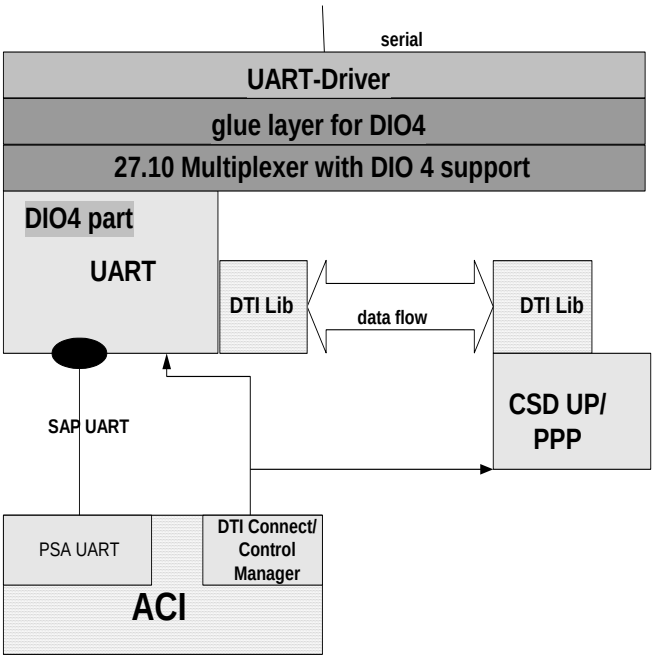

Figure 9: serial data and command flow within phase 1a at Helen1/Helen2

3.1.2.2 Packet Data

For the UART driver is needed a glue layer to DIO interface, vs.4. The entity PKTIO has to support DIO interface, vs.4 too. For translation from serial/packet to packet/serial data the multiplexer is necessary. The access to ACI is unchanged.

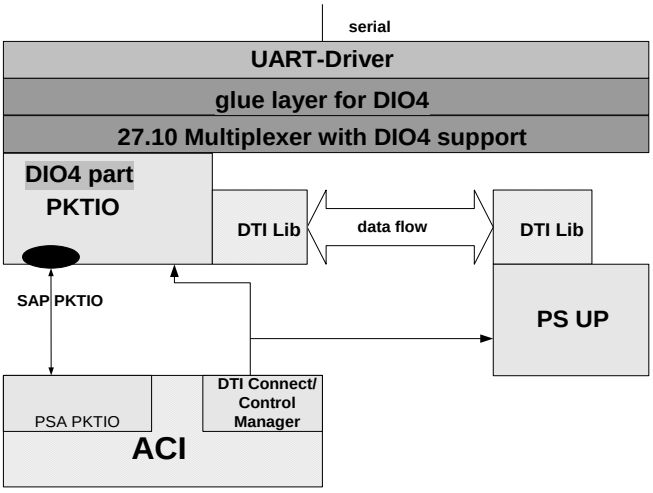

Figure 10: Packet data and command flow within phase 1a at Helen1/Helen2

3.1.3 Summary for Phase 1a

DIO Interface

- Specification, implementation and test of a interface which unifies all known requirements of existing interfaces and which is easy to extend to new interfaces. It has to process serial, packet and multiplexer data.

Multiplexer driver 27.010

- Specification, implementation and test of a new driver which provides the same multiplexer functionality as currently available with the UART entity
- Adaptation and test of this driver to the new DIO interface, version4.

PKTIO entity

- Adaptation and test of this entity to the new DIO interface, version4.

UART driver

- Adaptation and test of CSMI driver via new glue layer to the new DIO interface, version4.

UART entity

- Taking out the multiplexer functionality of this entity
- Adaptation and test this entity to the new DIO interface, version4.

3.2 Implementation Phase 1B

3.2.1 Calypso / Calypso+

The new entity PSI has to take over the access of the USB driver via the new DIO Interface, version 4 to the Protocol stack. For this reason SAP PSI is needed. ACI has to be extended with the appropriate counterpart. The behavior is similar to the current entity PKTIO.

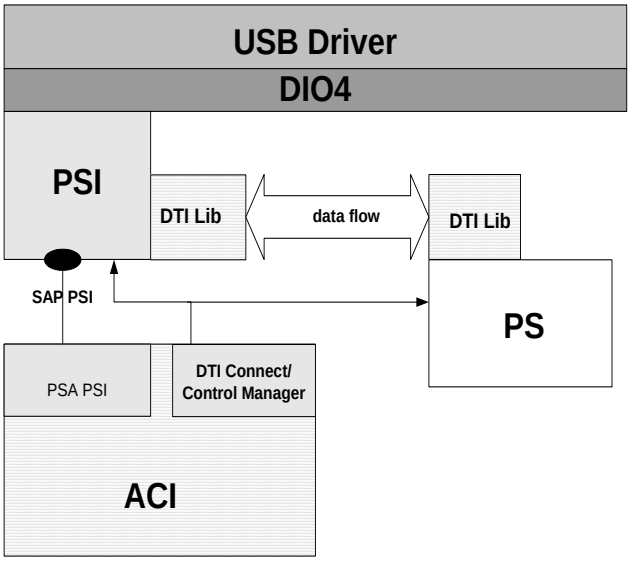

Figure 11: Serial/packet data and command flow within phase 1b at Calypso/Calypso+

3.2.2 Summary for Phase 1b

ACI

- Specification, implementation and test of new PSA PSI

PSI entity

- Specification, implementation and test of new entity PSI

USB driver

- Specification, implementation and test of new USB driver

In this phase the entities UART and PKTIO changed in phase 1a are used as described in chapter 3.1.

3.3 Implementation Phase 2

3.3.1 Perseus1/Perseus2

3.3.1.1 Serial Data

The new entity PSI has to take over the functionality of the UART entity. The SAP PSI replaces SAP UART.

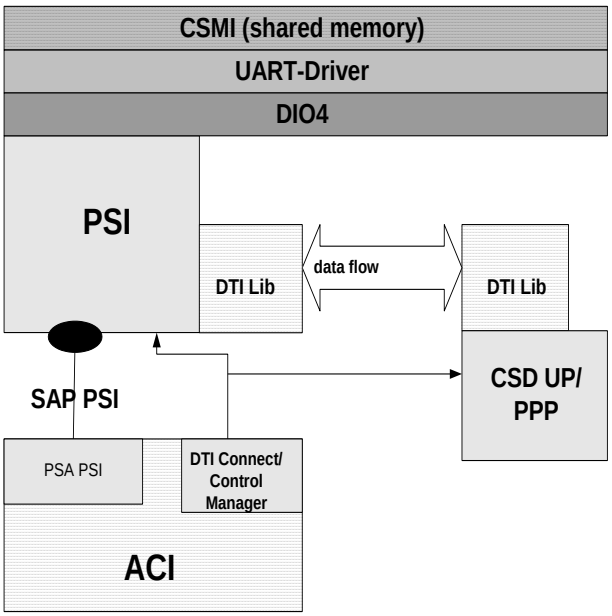

Figure 12: Serial data and command flow at Perseus1/Perseus2 within phase 2

3.3.1.2 Packet Data

The new entity PSI has to take over the functionality of the PKTIO entity. The SAP PSI replaces SAP PKTIO.

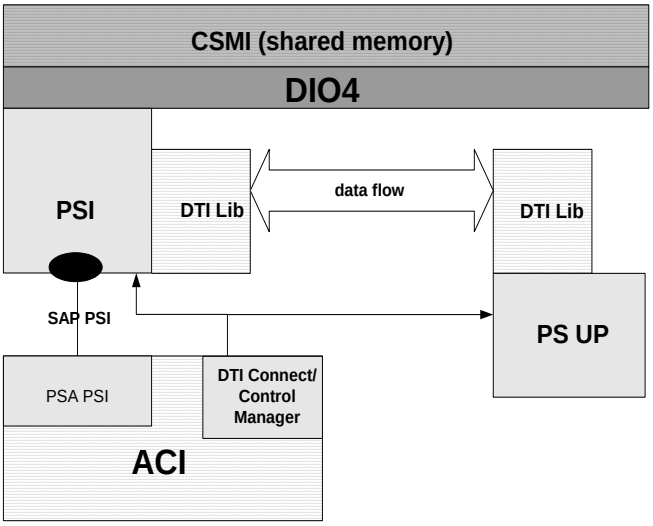

Figure 13: Packet data and command flow at Perseus1/Perseus2 within phase 2

3.3.2 Helen1/ Helen2

3.3.2.1 Serial Data

The new entity PSI has to take over the functionality of the UART entity. SAP PSI replaces SAP UART.

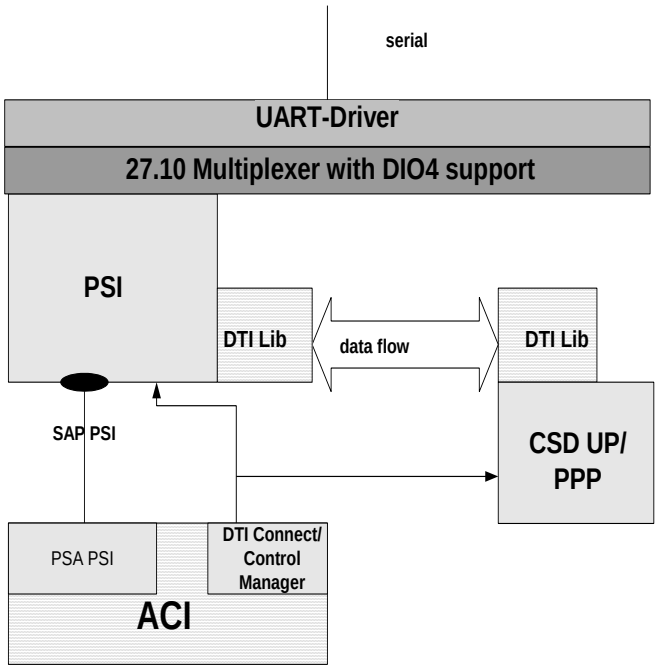

Figure 14: Serial data and command flow at Helen1/Helen2 within phase 2

3.3.2.2 Packet Data

The new entity PSI has to take over the functionality of the PKTIO entity. SAP PSI replaces SAP PKTIO.

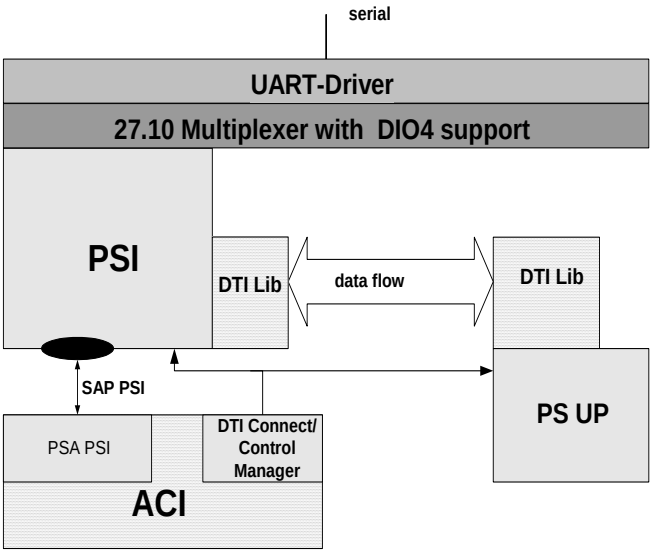

Figure 15: Packet data and command flow at Helen1/Helen2 within phase 2

3.3.3 Summary for Phase 2

ACI

- Taking out of the SAP's PKTIO, UART
- Adaptation of functionalities with respect to PKTIO and UART in ACI

PKTIO entity

- Removing this entity

UART entity

- Removing this entity

PSI entity

- Enhancement SAP PSI
- Providing functionality of UART entity, PKTIO entity
- Providing possibilities for simple enhancements of new driver requirements

Appendices

A. Acronyms

ACI	Application Control Interface
CSD UP	Circle Switched Data User Plane
DTI	Data Transmission Interface
DIO	Data I/O
PKTIO	Packet Data I/O
PPP	Point-To-Point-Protocol
PS UP	Packet Switched User Plane
PSI	Protocol Stack Interface
UART	Universal Asynchronous Receiver Transmitter

B. Glossary

device	A logical data connection which is handled by a driver.
driver	A driver has direct access to a hardware component and it provides an abstraction of that.