16

Think Systems - Software by LuK

Klaus Küpper
Roland Seebacher
Olaf Werner

Introduction

For over ten years LuK has not only been working in the conventional core business of mechanical components, but also in the field of automated transmission systems. Right from the start, in addition to mechanical components, the necessary automation strategies have been developed and put into practice.

Naturally, the question is raised and continues to be raised, as to whether LuK, as a classic metalworking automotive supplier, should concern itself with software development. However, experience has shown that the combination of components and software provides many advantages for the customer.

This paper highlights these advantages and explains the reasons why LuK undertakes software and system strategy development.

Furthermore, various principles of software development have evolved in recent years at LuK. Ultimately, they are simply an adaptation of those guidelines along which LuK has always developed its products. Hence the customer gets software of the quality he has grown to expect from LuK.

Selected examples illustrate how those 'theoretical' benefits and software development principles are transformed into reality at LuK.

Why Does LuK Develop Software?

Transfer of Drivetrain Knowledge

Over many years in its conventional clutch and dual mass flywheel business, LuK has placed great emphasis on not only offering the customer individual components, but optimising them within the whole field of the drivetrain.

This experience, which has been built up over many years, is reflected firstly in the components, but also in the optimisation of the entire drivetrain.

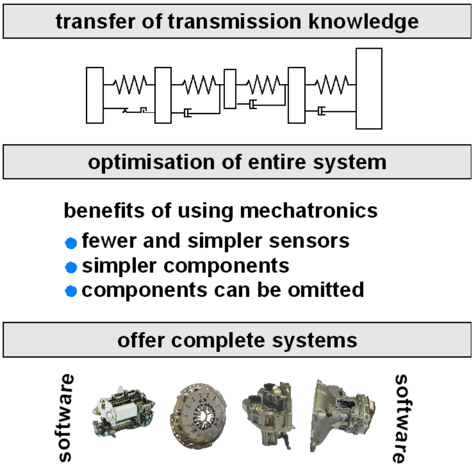

Fig. 1: *Customer Benefits of LuK Software*

It is exactly this experience that software development for XSG systems uses. In some respects software can be considered to be just another component of the transmission system which has to be tuned in combination with all the others. However, software also offers the possibility to actively (or 'intelligently') utilise all of LuK's experience, where the theoretically complex ideas could rarely be fully exploited in a purely mechanical solution.

The customer therefore has the opportunity of acquiring know-how (strategies and ideas) straight from LuK and applying it directly to his vehicle. It is critical that knowledge about the clutch, which defines drivetrain comfort during slip phases, is fully utilised in the software. And it is precisely in the field of clutches that LuK can boast of long and extensive experience.

Optimisation of the Entire System

Only by applying electronics and software do the mechanics become a 'mechatronic system'. This catchphrase, which has been widely used in recent years, really only means that instead of mechanics, electronics and software being considered separately, they are viewed as an integrated concept, whose optimisation exceeds the boundaries of individual components. Hence, really intelligent system strategies enable the saving of certain sensors, such as, for example, the gearbox input shaft speed sensor and the release travel sensor in the clutch bell housing. Software makes the manufacturing tolerances of the manual gearbox controllable. Intelligent algorithms learn these tolerances at the end of line, so that they can then be taken into account in the subsequent control operation. Finally, only by considering the entire system can certain software requirements be transferred to the mechanics. One example of this is the adaptation of the clutch to the requirements of the control strategy.

In addition to improving the entire system itself, the customer can naturally also look forward to optimum integration, simply by using the LuK drivetrain knowledge mentioned above.

Offering Complete Systems

LuK is working on various concepts in the field of drivetrain automation, based on a gearbox with spur gears (XSG), which will be presented in other symposium papers. They are all characterised by the fact that they include a basic gearbox, a clutch, a clutch actuator, a gear actuator, and electronics. Every individual component fulfils a specific task for which it is optimised.

It is only when they are combined with the aid and co-ordination of software, that these individual components form a system, see figure 2.

By developing strategies and converting them into software, LuK is able to offer the customer complete drivetrain automation instead of individual components. Hence the customer needn't worry about the co-ordination and integration of sub-components, but can instead concentrate on optimising the system to the vehicle.

Also this transition from component to system suppliers – a still quite strong trend in the automotive industry – is a reason for LuK's inhouse software development.

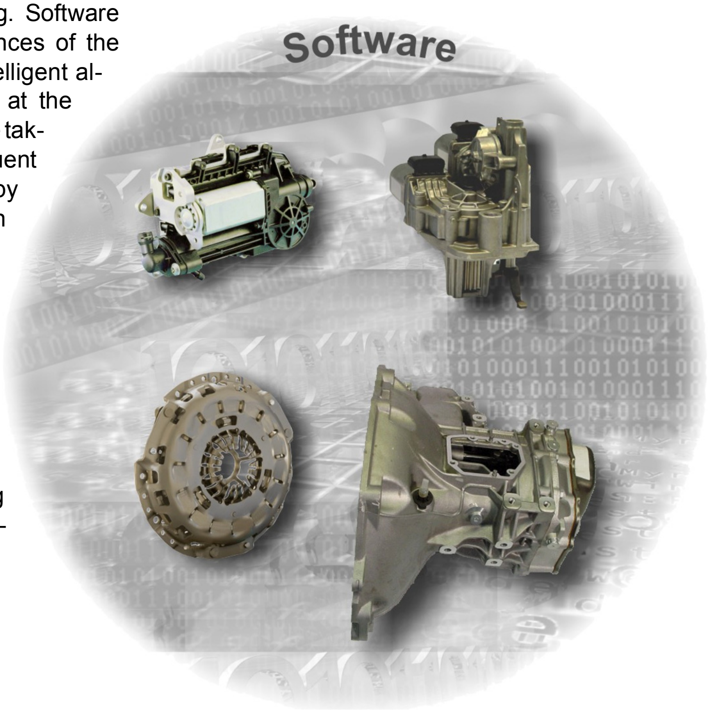

Fig. 2: *Complete System by Means of Software*

Principles of Software Development

Over the years various principles of software development have been established at LuK. Obviously, every piece of software must fulfil a multitude of requirements – just like every mechanical component. However, the principles described in the following are of particular importance; they are the benchmarks for each new strategy.

It is interesting that these principles are equally applicable to the development of mechanical components and play a similarly important role there.

System Knowledge

As stated above, software should combine all individual components into an entire XSG system. This again relates to the entire vehicle context or its drivetrain. It naturally follows that the person who writes the software must not only understand all individual components and how they interact, but also the primary system 'drivetrain' in detail. Finally, the degree of system knowledge determines the quality of the strategies and software.

Besides this software development principle forming a clear part of the LuK method of working, it is also evident in the calibre of the employees. The software development team at LuK is composed equally of mechanical engineers, electrical engineers, and physicists. Hence the focus is on the scientific understanding of problems, leading to the best, theoretically well-founded solutions.

system knowledge

robustness

safety

Fig. 3: *Principles of Software Development*

Robustness

For a long time, especially since the spread of Windows computers, software has been associated with a certain unreliability. This is completely unacceptable in the automotive field. The second important principle of software development at LuK is therefore robustness. At the end of the day, software must function in all vehicles and with all components over the entire service life and in every situation.

Firstly, this means that the software itself must not contain any error which could lead to a malfunction of the system.

Furthermore, the system strategies must be designed so that they function perfectly, both with all tolerances as well as with all other variations, e.g. frictional or power output changes in motors. Even changes during the service life have to be taken into consideration.

Finally, the software must function correctly in every unexpected situation, something which also requires careful consideration.

However, as with every technical application, a high degree of robustness may occasionally require compromises with optimum functionality be made. It is critical that it does not cause inconvenience to the driver.

An example of this is the start-up, which should ideally always occur at the same engine speed.

A strategy which usually provides a precise engine speed, but which, in the event of a build-up of unfavourable conditions, might cause a deviation of 500 min^{-1} or even engine speed oscillations, would not be acceptable. A robust strategy that does not behave like this is therefore chosen. Whilst this strategy may still deviate from the ideal value by, say, 100 min^{-1} among different vehicles in various situations, the driver would not generally notice this.

As in every aspect of engineering, a compromise must be found between accuracy and robustness, which, in the case of doubt, falls in favour of robustness.

Safety

Specifically in the automation of the drivetrain, safety plays a primary role. Safety here not only relates to actual software quality, but also to functionality. There is no place for compromise. Every function is therefore carefully tested in terms of safety-relevant issues. Our customers know just how much value LuK places on this rule.

This practice will continue in the future.

Slip Control

Slip control is a typical area in which only software enables an idea to be realised.

The basic idea of slip control, isolating the drivetrain against vibrations of the engine through a slightly slipping clutch, is not a new one and has been repeatedly discussed over recent years. Apart from various mechanical problems in the past, insufficient opportunities to control the clutch have prevented the use of slip control first and foremost.

The vibrational principles and the design of the drivetrain for slip control will not be examined closely here, as these issues were already dealt with in detail at the LuK Symposium in 1998 [1].

Control Concept

The first task of control is to define an optimum desired slip. This represents a compromise of necessary vibration isolation and permissible increase in wear or fuel consumption. It is precisely here that extensive tests and simulations, in particular vehicle-specific, must be conducted.

The second key point is to set the calculated desired slip as precisely as possible. A high level of control accuracy is necessary here, since if the slip deviates downwards, this can lead to sticking and thus to acoustic problems (booming); however, for reasons of wear and fuel consumption, neither should it deviate upwards.

In order to achieve the required robustness of software strategy mentioned in the previous section, LuK is pursuing an approach which combines open-loop control, adaptation and closed-loop control. This combination characterises many LuK strategies and will be explained in more detail in the slip control example.

In terms of definition, a closed-loop controller always feeds back the measured output of the controlled system (plant), in order to generate a control input for the plant such that the ‘error’ is minimised (c.f. figure 4a).

Open-loop control, on the other hand, merely uses the desired value as an input and – whilst knowing the system to be controlled – generates a control input, which, in the ideal scenario, correctly sets the desired value (c.f. figure 4b).

For the moment, a potential error between reference signal and output is not corrected here.

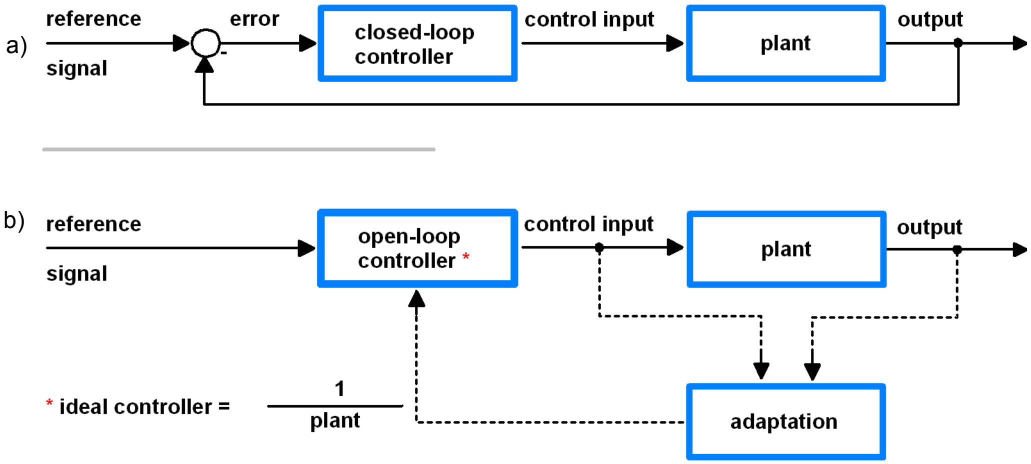

Fig. 4: *a) Closed-loop Control*
b) Open-loop Control

Nevertheless, in order to achieve good quality, the behaviour of the plant is observed – in the case in question, the entire vehicle with engine, clutch and drivetrain. Using this information, the model of the controlled system, which the open-loop control uses internally, is adapted to compensate for medium and long-term plant changes.

Both methods have pros and cons, which is why in practice they are generally used in parallel.

LuK is pursuing the approach of using open-loop control as much as possible and closed-loop control as little as necessary. Using a small proportion of closed-loop control should achieve the following in particular:

- The closed-loop controller will not become unstable in any tolerance positions or environmental situations.
- Oscillations or overshoots by the closed-loop controller won't be noticeable.
- The speed advantage of the open-loop control is exploited.

Obviously, at the same time, attention must be paid to the fact that as much system knowledge as possible should be used, so that the open-loop control only permits a minor error.

In the actual case of slip control, the system being controlled essentially consists of clutch, engine and drivetrain. The driver command acts as an input on the engine. The slip control acts on the clutch.

In order to achieve an already precise feed-forward control, the slip controller receives various other information in addition to the driver command, in particular about the engine condition. This is used in a model of the plant to calculate the necessary control input correctly.

Partial Slip

Extensive investigations at LuK have shown that sufficient vibration isolation is already achieved in partial slip, as shown in figure 5. Partial slip is characterised by the fact that due to the rotational irregularity of the combustion engine, the sticking and slip phases alternate with the ignition frequency.

For reasons of wear and fuel consumption, as low a desired slip as possible is sought during slip control. Its lower limit is pre-determined by the necessary vibrational isolation and the control accuracy. Continuous sticking should thus be avoided in every case. It is precisely this what has previously been viewed as extremely difficult, and low slip has therefore been widely avoided. LuK's investigations, however, paint a completely different picture:

Firstly, the situation during partial slip is analysed more closely (c.f. figure 5). The simulation shows that in the torque peaks, hence with greater forward acceleration of the combustion engine, the clutch starts slipping. During slip, the transferred torque is limited to the maximum transferable clutch torque. As soon as the rotational irregularity of the engine leads to a sticking clutch, the torque transferred into the transmission drops from the transferable clutch torque to the internal torque. This process, which is easy to comprehend, has the effect that the average transferred torque is below the transferable clutch torque.

The average clutch torque transferred in partial slip is determined by the ratio of slip to continuous sticking and the torque irregularity of the engine. This results in the physical behaviour shown in figure 6: from slip speed zero to transition into full slip (the engine irregularities no longer lead to a temporary sticking of the clutch), the average transferred clutch torque rises constantly. It achieves the actual transferable torque in the traditional sense during transition to full slip.

If this torque is interpreted as a friction co-efficient, a positive 'friction co-efficient gradient' can be assumed in partial slip, whereby the gradient increases at small slip speeds. A potential negative friction co-efficient gradient in the true sense is even compensated for by the effect described.

This important discovery of the positive 'friction co-efficient gradient' in partial slip holds a considerable advantage for slip control.

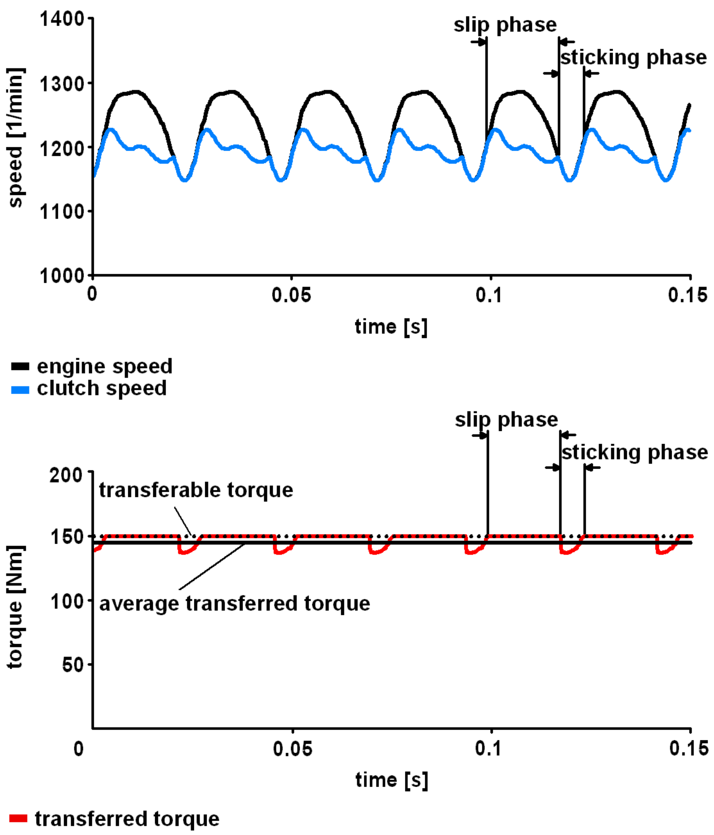

Fig. 5: *Partial Slip*

In contrast to the dreaded negative friction co-efficient with clutch linings, a positive friction co-efficient gradient does not lead to juddering, but rather independently stabilises the slip speed initially set: as soon as the slip increases through a slightly higher engine torque, the torque transferred by the clutch increases, whereupon the slip again reduces.

This self-stabilising effect also allows the use of a more agressive closed-loop gain in partial slip, without risking instability. It is thus possible to ensure a particularly high level of controller precision and a fast response to errors or torque changes.

Due to the increasingly positive 'friction co-efficient gradient', control accuracy can be optimised so that permanent sticking can continue to be safely avoided.

Only the careful analysis of behaviour during partial slip and the subsequent use of this discovery enabled LuK to demonstrate slip control with high accuracy and low slip speeds.

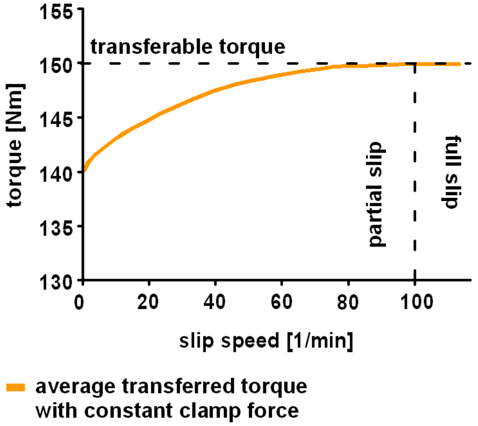

Fig. 6: *Slip Dependency of the Transferred Torque*

Results

The combination of open-loop control and closed-loop control during partial slip leads to a stable system strategy that also delivers very good results.

Figure 7 shows a typical measurement. The top diagram shows the course of engine and gearbox input speed. It clearly shows how the slip speed is set during a back-out, when coasting, during a tip-in, and during the subsequent acceleration.

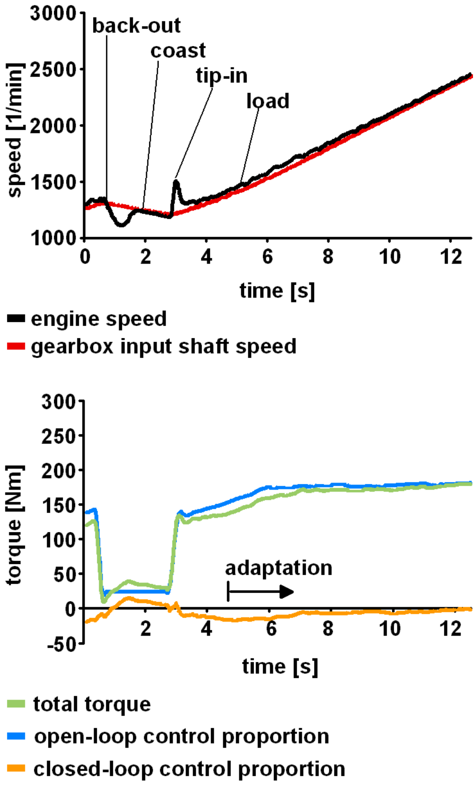

Fig. 7: *Vehicle Measurement Slip Control*

The deviations in speed during large load changes prevent transmission vibrations without being acoustically noticed by the driver. Comfort during load changes can thus be increased without the responsiveness of the vehicle being impaired.

According to the slip characteristic map the desired slip is continually reduced with increasing engine speed. As soon as the defined engine speed is achieved and there is no slip, the clutch torque is increased to higher load.

The lower part of the diagram shows how the desired clutch torque consists of an open-loop control dependent part and a closed-loop control part. In accordance with the structure selected, the open-loop control proportion is clearly predominate. Furthermore, it can be seen how, after 5 seconds, the closed-loop control proportion becomes continually smaller in favour of the open-loop control proportion. The adaptation here ensures that the open-loop control precision increases and the closed-loop control proportion drops further.

The slip control described here has already been used in a vehicle – a Mercedes A-class A170 modified to a single mass flywheel, which was in an endurance run. It shows that a vibration isolation is achieved comparable with a dual mass flywheel (DMF). At the same time, the control was stable and reliable over the endurance test. The extra wear on the clutch lining is only 0.2 to 0.4 mm per 100 000 km. Fuel consumption essentially remained unchanged, which was also demonstrated by driving cycle simulations. This can be explained by the fact that the engine inertia was able to be reduced through the omission of the DMF. The reduction in acceleration losses thus achieved compensates for the minimal extra fuel consumption caused by slip. The results must naturally be verified for actual vehicle projects, each according to the required and necessary driving and desired slip profile.

Clutch Protection

Obviously, when used as a start-up element, the clutch can be overloaded due to driver misuse. The robustness demanded for the software means that it protects the system better in this situation than with a manual transmission.

A key example is driving or start-up on a hill. During a typical hill start-up with automated clutch actuation, firstly the engine speed slightly increases and the clutch slips. Only when a start-up speed that corresponds to the vehicle speed has been reached, will the clutch close completely. However, if the vehicle is driven uphill relatively slowly then the engine and clutch speeds may never synchronise. To avoid this constant slip, clutch control imitates the ‘strategy’ of a normal driver:

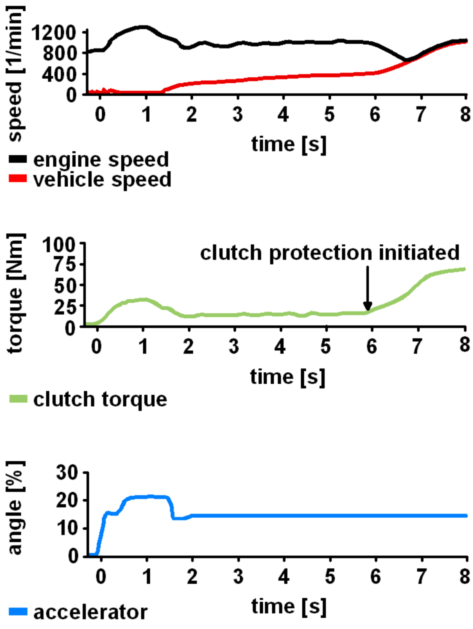

Fig. 8: *Clutch Protection: Very Slow Driving*

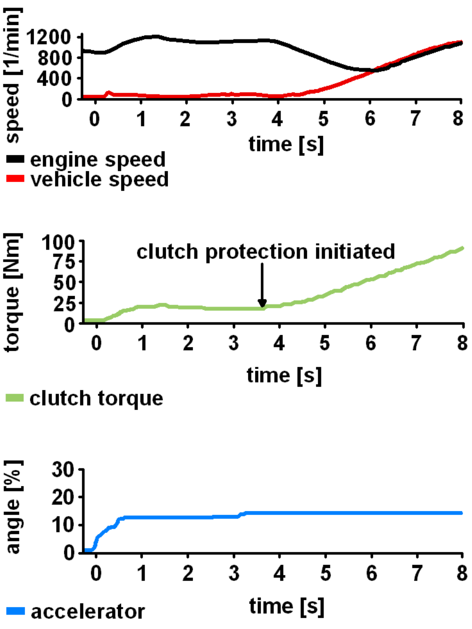

Fig. 9: *Clutch Protection: Holding on a Hill*

As soon as the vehicle is moving, the clutch is slowly closed completely. This means that the vehicle drives uphill at low speed without loading the clutch. This strategy is shown in figure 8.

Another, particularly critical scenario would be if the vehicle was held on the hill by modulating the accelerator, as might occur in a traffic jam on a hill for example. A driver with a foot-operated clutch will not usually behave like this, because modulation via acceleration and clutch pedal is relatively difficult here. On the other hand, an automated clutch facilitates hill-holding.

A strategy for preventing misuse of the clutch is illustrated in figure 9. After a certain time, approximately 4 seconds, the clutch is gradually closed. Since the driver operates the accelerator and hence wishes to start up, and clutch torque is built up very slowly, the vehicle response comes as no surprise to the driver; he can respond appropriately: either by continuing the start-up or holding the vehicle on the hill by braking. He simultaneously experiences and ‘learns’ the natural limits of the system.

Both these examples demonstrate how, through appropriate, relatively simple software measures, the robustness of the entire system can be substantially increased against misuse.

Monitoring Concept

The processor monitoring system described below can serve as an example of the safety considerations necessitated by a shift-by-wire system, such as an ASG or another XSG system.

The micro-processor is a central component in all XSG processes, be it gear selection, engine intervention or clutch control. For this reason, its monitoring has a special significance, since a single error can lead to dangerous situations. In order to detect these situations and render it safe, a multi-level monitoring concept, the so-called ‘Intelligent Safety Monitoring’ (ISM), was developed.

The ISM mainly considers a series of particularly critical situations, so-called ‘top events’. It is their occurrence, as the result of a processor error, that it must safely prevent.

Safe Condition

The first question which is raised when developing a monitoring concept regards the safe condition to which the system should be transferred. If it is assumed, as is necessary here, that there is a processor error, there are only very limited response possibilities in the event of a failure. Since the ‘intelligence’ of the system has possibly failed, it is very difficult or even impossible to respond to different error situations with various, perhaps even complex, strategies.

Figure 10 illustrates considerations on a safe condition for the clutch, which must be regarded as the most important power transfer element of the transmission system.

A clutch actuator can fundamentally ***close***, ***open*** or ***stop*** in the present position as a potential response to an error. Other strategies, such as various time sequences of opening and closing, perhaps even as a response to external signals, are barely conceivable during a processor failure.

For potential initial situations, i.e. ***clutch closed***, ***clutch in an intermediate position*** or ***clutch open***, it should be considered which of the three responses mentioned above is the safest.

If the clutch is closed, for example as a roll-off lock (top event: vehicle is on a hill with engine stopped and gear engaged), closure or remaining closed is the safest option. On the other hand, opening the clutch can lead to an unexpected response for the driver, i.e. the vehicle suddenly rolling away.

On the other hand, if the clutch is open, then opening or remaining open is the safe option. Closure that is totally unexpected by the driver, for example at a zebra crossing or traffic lights (top event), might have fatal consequences.

If the clutch is in an intermediate position, further opening or closing can either be safe or lead to very critical situations, depending on the actual situation. Stopping the clutch actuator is the safest response here, since the driver will not be confronted with an unexpected vehicle action.

It is clear that stopping the clutch actuator in all situations represents the safest option. The advantage of the electro-mechanical clutch actuator favoured by LuK is also shown here, namely, that when switching off the actuator or power supply, the position remains unchanged and the clutch does not close or open, as on a hydraulic system.

		situation		
		closed	**intermediate position**	**open**
response	**close**	✓	?	⚡
	open	⚡	?	✓
	stop	✓	✓	✓

Fig. 10: *Safe Condition during Total Failure*

ISM

To fulfil the requirements of a shift-by-wire system, a monitoring structure with several levels was developed jointly with Bosch. It is fundamentally based on the fact that various software levels, as well as the main processor used and a monitoring processor control each other. Figure 11 provides an explanation of the system.

The largest part of the monitoring runs at the various software levels in the main processor. Consequently, various monitoring levels are

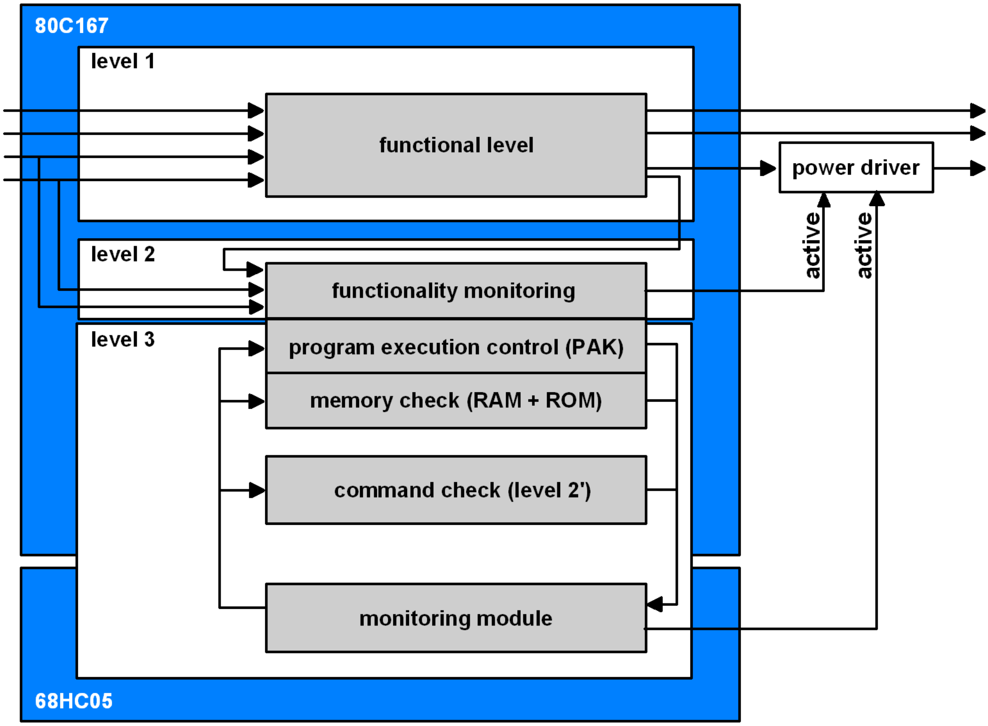

Fig. 11: *Structure of 'Intelligent Safety Monitoring'*

implemented here in addition to the actual functionality for ASG control (functional level). The main processor (an Infineon 80C167 in the present ASG) and a monitoring computer (a Motorola 68HC05 in the present ASG system) are used as hardware.

Each of these computers can independently switch off the power drivers of the XSG system if necessary.

Furthermore, they can trigger a reset on the control unit in order to re-initialise the system in the event of a processor failure having occurred and enable driving to continue.

Level 1 (functional level) contains the complete functionality in normal and limp-home operation, that is to say clutch control, gear selection, or engine intervention.

Level 2 (functionality monitoring) monitors Level 1. For this, the input signals from Level 1 and the output signals generated by it are read in and checked for plausibility. However, it is not the entire functionality of Level 1 that is covered, but only the safety critical functions (top events) that are monitored.

A simple example is the prevention of clutch closure with an engaged gear, without the driver operating the accelerator. Independently of all other computations, error monitoring or adaptations that the functional level must conduct, functionality monitoring merely needs to ensure that the clutch does not close beyond the creeping torque. As soon as this relatively simple rule is violated, the functionality monitor traces this back to a controller error and switches off the power drivers.

Level 3 covers various tests which again ensure the correct functioning of monitoring Level 2. This includes a program execution control, which ensures that every part of functionality monitoring is carried out, as well as continuous memory check of RAM and ROM. Command check for Level 2 is also executed.

This means that all relevant processor commands that Level 2 uses are checked for correct functioning. This extensive monitoring of Level 2 by Level 3 can only be realised, because Level 2 includes substantially less code compared to Level 1.

The last module in the monitoring process is the monitoring module. With the help of randomly selected ‘questions’ to Level 3 of the main processor and the answers generated from the software modules of Level 3, the monitoring computer checks their integrity. The main controller also uses this exchange of questions and answers in reverse, to monitor the second processor. The monitoring processor must therefore detect intentional ‘false’ answers thrown in and verify these with the main processor.

If a safety critical error is detected in one of the three levels of the monitoring concept, the power drivers are switched off and a reset of the computer is triggered. By switching off the power drivers, the XSG system goes into the safe condition defined above and can rectify itself again if the controller error is only temporary. Otherwise, the system remains in safe condition, until the error is rectified.

The safety structure described is heavily based on designs which are also used with other safety relevant components in the drive train, for example, on electronic throttle control systems.

Summary

More than ten years of software development at LuK have shown that the extension of LuK expertise beyond mechanical components offers considerable advantages for the customer. It enables LuK to

1. transfer their transmission system knowledge, acquired over many years, beyond the limits of mechanical hardware alone into software strategies
2. optimise the automated transmission system as a whole in the vehicle and
3. offer the customer complete systems for automating transmission systems.

LuK is thus pursuing the development of entire system knowledge, maximum robustness and uncompromised safety as the most important criteria for software development.

This paper illustrates typical examples of the advantages for customers and for the development guidelines, namely: slip control, which has already shown some very good practical results; some strategies for clutch protection; and a safety monitoring concept that makes the shift-by-wire system acceptable in vehicles.

Literature

[1] Fischer, R.; Berger, R.: Automation of Manual Transmissions, 6th LuK Symposium 1998.