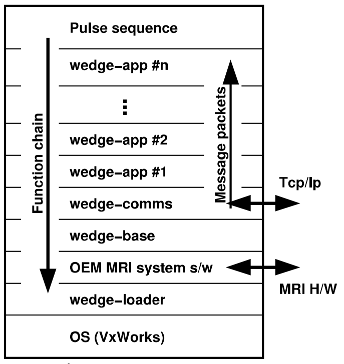
Modular extensions to MRI systems architecture with example application of pulse sequence independent real-time scan plane control

J. Andrew Derbyshire[1] and Peter A. Bandettini[1]
[1]*fMRI Core, National Institute of Mental Health, NIH, Bethesda, MD, United States*

Introduction: This work describes an approach for extending the capabilities and features of MRI systems in directions that are outside the scope of the standard pulse sequence development environments. These extensions are sought in order to support novel applications such as very lengthy scans e.g. for resting state or sleep fMRI. For example allowing the system to interact or synchronize with external (e.g. fMRI stimulation) equipment (e.g. for fMRI stimulation or physiological monitoring). Other applications include biofeedback, self-tuning MRI and real-time shimming. In this report, we demonstrate the ability to control the MR system with a pulse-sequence independent implementation of a real-time scan plane control system.



Theory: The real-time control platform at the heart of many MRI systems is based on the use of the VxWorks (Wind River, Alameda, CA) operating system. VxWorks permits the linking of software objects into a running system. Our approach is to insert VxWorks software modules (referred to as *wedges*) into the MRI system software architecture so that our software wedges reside between the MRI system hardware drivers responsible for actuating the various MRI RF and gradient waveforms and the main pulse sequence in the MRI software stack (see Figure 1).

The software modules are loaded in sequential order during the boot process of the MRI scanner, with the VxWorks OS resolving the linking between the various functions in different modules. By inserting the software modules in a specific order, it is possible to intercept calls between the pulse sequence and the underlying hardware drivers routines, permitting the software in the *wedge* to briefly gain control of the execution thread at specific times during the pulse sequence operation, such as pulse sequence downloading, the sending of sequence frames (TRs), and the completion of scans. At these times, it is then possible to examine (and optionally modify) the pulse sequence waveforms, or modify other aspects of the system such as center frequency, shims, eddy current correction, etc.

Methods: Implementations of the software stack were created for GE Signa HDx and MR750 3T MRI systems (Waukesha, WI, USA). Software wedge modules were created using GCC C and C and C++ language cross compilers (Free Software Foundation, Boston, MA) for VxWorks PowerPC architectures. A *wedge loader* module was created and substituted for the OEM scanner software so that it would automatically be started when the MRI system's real-time computer was booted. The loader's first action is to load the OEM software module and it then, in-turn, loads further wedge modules according to an easily modifiable configuration file, which determines exactly which software modules are required, and their order in the software stack. Upon loading a wedge, the VxWorks OS links it with previously loaded software modules and resolves undefined symbols. The newly inserted wedge is then allowed the opportunity to initialize itself either by calling an initialization function, or the starting of a dedicated thread (with a sufficiently low priority such that it doesn't block the main pulse sequence task) for that wedge.

The base level wedge, and other application wedges deliberately alias some of the MRI system function routines, effectively substituting the wedge implementation of the function in place of the OEM call. In effect, this establishes function call-chains, in which key MRI system calls from the pulse sequence cascade down the chain as they are passed from wedge-to-wedge for processing, before finally being received by the OEM MRI system. This gives each wedge an opportunity to make particular adjustments to the system at those times specifically associated with that particular call, such as a sequence download, modification of a waveform or the readiness of the next TR of the sequence.

One particular wedge is the TCP/IP communication module, which creates and listens on a socket to allow external computers to connect with the running MR system. The communications wedge similarly establishes a communications packet function call-chain to which other wedges can attach. When the communications module receives a control data packet from the external workstation, the packet is passed along to each registered wedge in the chain.

Typically a wedge is associated with the implementation of a particular function. Wedges that are not required for a particular application can be temporarily deactivated, effectively removing them from the various function call-chains, but remain loaded until they are needed again.

Example: Pulse sequence independent real-time scan plane control. As a motivational example, we describe a wedge that permits interactive control of the scan plane. While real-time scan plane control has been previously demonstrated, and is already a feature on some scanners, it is typically only available for a very limited number of sequences. Furthermore, the implementation at the pulse sequence level requires the tedious modification of all the RF pulse and readout phase and frequency settings in that sequence. As a demonstration system, we have developed a system that permits the vast majority of the pulses sequences to be interactively controlled without having to explicitly modify each sequence.

The scan-plane tracking wedge is controlled by an external workstation that sends message packets that contain a rigid body (rotation+translation) transformation matrix. The wedge intercepts the pulse sequence thread when each new TR of the scan has been prepared. The sequence instructions and waveforms for the upcoming TR are examined. The position and orientation of the slice and readout are determined from the frequency and gradients employed in the sequence, and the position of the slice is recalculated using the transformation matrix, and then the sequence instructions/waveforms and rotation matrix are updated accordingly. For sequences with readout, updating the acquisition phase requires careful subtraction of only the previously applied receiver phase associated with phase-encoding so that any RF phase cycling (e.g. balanced SSFP, RF spoiling, EXORCIST, etc.) is retained. The system can also be applied to preparation sequences and the system prescan.

Conclusions: We have demonstrated methods for extending the MRI system software capabilities in directions different to those normally associated with the MR pulse sequence development environment. This provides a very modular and flexible environment for researchers looking to control or synchronize the scanner with external systems for novel applications. The system was demonstrated with via a powerful real-time scan plane control mechanism that allows the control of the vast majority of pulse sequences without having to explicitly create an interactive version of each sequence. These methods could, for example, be very useful in combination with a camera system for (e.g.) [1, 2] prospective motion correction.

References: [1] Qin et al, MRM, **62**:924, 2009, [2] Zaitsev et al., Neuroimage, **31**:1038, 2006.