Software Manual

R&S® TSMW Interface & Programming Manual

R&S® TSMW-K1

1503.3960.02

Printed in Germany

ROHDE&SCHWARZ
Test & Measurement

1503.3776.32-02

Dear Customer,
The R&S® TSMW-K1 makes use of open source software packages. The most important of them are listed below together with their corresponding open source license. The verbatim license texts are provided in on the user documentation CD-ROM (included in delivery).

Package	Link	License
OpenSSL	http://www.openssl.org/	OpenSSL / SSLeay

The R&S® TSMW-K1 includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit ("http://www.openssl.org/"). It includes cryptographic software written by Eric Young (eay@cryptsoft.com) and software written by Tim Hudson (tjh@cryptsoft.com).

Throughout this manual, TSMW-K1 is generally used as an abbreviation for the R&S® TSMW-K1.
Throughout this manual, TSMW is generally used as an abbreviation for the Radio Network Analyzer R&S® TSMW.

1503.3776.32-02

Grouped Safety Messages

Make sure to read through and observe the following safety instructions!

All plants and locations of the Rohde & Schwarz group of companies make every effort to keep the safety standard of our products up to date and to offer our customers the highest possible degree of safety. Our products and the auxiliary equipment required for them are designed and tested in accordance with the relevant safety standards. Compliance with these standards is continuously monitored by our quality assurance system. The product described here has been designed and tested in accordance with the EC Certificate of Conformity and has left the manufacturer’s plant in a condition fully complying with safety standards. To maintain this condition and to ensure safe operation, observe all instructions and warnings provided in this manual. If you have any questions regarding these safety instructions, the Rohde & Schwarz group of companies will be happy to answer them.

Furthermore, it is your responsibility to use the product in an appropriate manner. This product is designed for use solely in industrial and laboratory environments or, if expressly permitted, also in the field and must not be used in any way that may cause personal injury or property damage. You are responsible if the product is used for an intention other than its designated purpose or in disregard of the manufacturer's instructions. The manufacturer shall assume no responsibility for such use of the product.

The product is used for its designated purpose if it is used in accordance with its product documentation and within its performance limits (see data sheet, documentation, the following safety instructions). Using the product requires technical skills and a basic knowledge of English. It is therefore essential that only skilled and specialized staff or thoroughly trained personnel with the required skills be allowed to use the product. If personal safety gear is required for using Rohde & Schwarz products, this will be indicated at the appropriate place in the product documentation. Keep the basic safety instructions and the product documentation in a safe place and pass them on to the subsequent users.

Symbols and safety labels

⚠	⚠ 18 kg	⚡	♨	⏚	⏚	⏚	⚠
Observe product documentation	Weight indication for units >18 kg	Danger of electric shock	Warning! Hot surface	PE terminal	Ground	Ground terminal	Attention! Electrostatic sensitive devices

I ○	⏻	⎓	∼	≂	⧈
Supply voltage ON/OFF	Standby indication	Direct current (DC)	Alternating current (AC)	Direct/alternating current (DC/AC)	Device fully protected by double/reinforced insulation

Observing the safety instructions will help prevent personal injury or damage of any kind caused by dangerous situations. Therefore, carefully read through and adhere to the following safety instructions before putting the product into operation. It is also absolutely essential to observe the additional safety instructions on personal safety that appear in relevant parts of the product documentation. In these safety instructions, the word "product" refers to all merchandise sold and distributed by the Rohde & Schwarz group of companies, including instruments, systems and all accessories.

Tags and their meaning

DANGER DANGER indicates a hazardous situation which, if not avoided, will result in death or serious injury.

WARNING WARNING indicates a hazardous situation which, if not avoided, could result in death or serious injury.

CAUTION CAUTION indicates a hazardous situation which, if not avoided, may result in minor or moderate injury.

NOTICE NOTICE indicates a property damage message.

In the product documentation, the word ATTENTION is used synonymously.

These tags are in accordance with the standard definition for civil applications in the European Economic Area. Definitions that deviate from the standard definition may also exist in other economic areas or military applications. It is therefore essential to make sure that the tags described here are always used only in connection with the related product documentation and the related product. The use of tags in connection with unrelated products or documentation can result in misinterpretation and thus contribute to personal injury or material damage.

Basic safety instructions

1. The product may be operated only under the operating conditions and in the positions specified by the manufacturer. Its ventilation must not be obstructed during operation. Unless otherwise specified, the following requirements apply to Rohde & Schwarz products: prescribed operating position is always with the housing floor facing down, IP protection 2X, pollution severity 2, overvoltage category 2, use only in enclosed spaces, max. operation altitude 2000 m above sea level, max. transport altitude 4500 m above sea level.
   A tolerance of ±10% shall apply to the nominal voltage and of ±5% to the nominal frequency.
2. Applicable local or national safety regulations and rules for the prevention of accidents must be observed in all work performed. The product may be opened only by authorized, specially trained personnel. Prior to performing any work on the product or opening the product, the product must be disconnected from the supply network. Any adjustments, replacements of parts, maintenance or repair must be carried out only by technical personnel authorized by Rohde & Schwarz. Only original parts may be used for replacing parts relevant to safety (e.g. power switches, power transformers, fuses). A safety test must always be performed after parts relevant to safety have been replaced (visual inspection, PE conductor test, insulation resistance measurement, leakage current measurement, functional test).

3. As with all industrially manufactured goods, the use of substances that induce an allergic reaction (allergens, e.g. nickel) such as aluminum cannot be generally excluded. If you develop an allergic reaction (such as a skin rash, frequent sneezing, red eyes or respiratory difficulties), consult a physician immediately to determine the cause.
4. If products/components are mechanically and/or thermically processed in a manner that goes beyond their intended use, hazardous substances (heavy-metal dust such as lead, beryllium, nickel) may be released. For this reason, the product may only be disassembled, e.g. for disposal purposes, by specially trained personnel. Improper disassembly may be hazardous to your health. National waste disposal regulations must be observed.
5. If handling the product yields hazardous substances or fuels that must be disposed of in a special way, e.g. coolants or engine oils that must be replenished regularly, the safety instructions of the manufacturer of the hazardous substances or fuels and the applicable regional waste disposal regulations must be observed. Also observe the relevant safety instructions in the product documentation.
6. Depending on the function, certain products such as RF radio equipment can produce an elevated level of electromagnetic radiation. Considering that unborn life requires increased protection, pregnant women should be protected by appropriate measures. Persons with pacemakers may also be endangered by electromagnetic radiation. The employer/operator is required to assess workplaces where there is a special risk of exposure to radiation and, if necessary, take measures to avert the danger.
7. Operating the products requires special training and intense concentration. Make certain that persons who use the products are physically, mentally and emotionally fit enough to handle operating the products; otherwise injuries or material damage may occur. It is the responsibility of the employer to select suitable personnel for operating the products.
8. Prior to switching on the product, it must be ensured that the nominal voltage setting on the product matches the nominal voltage of the AC supply network. If a different voltage is to be set, the power fuse of the product may have to be changed accordingly.
9. In the case of products of safety class I with movable power cord and connector, operation is permitted only on sockets with earthing contact and protective earth connection.
10. Intentionally breaking the protective earth connection either in the feed line or in the product itself is not permitted. Doing so can result in the danger of an electric shock from the product. If extension cords or connector strips are implemented, they must be checked on a regular basis to ensure that they are safe to use.
11. If the product has no power switch for disconnection from the AC supply, the plug of the connecting cable is regarded as the disconnecting device. In such cases, it must be ensured that the power plug is easily reachable and accessible at all times (corresponding to the length of connecting cable, approx. 2 m). Functional or electronic switches are not suitable for providing disconnection from the AC supply. If products without power switches are integrated in racks or systems, a disconnecting device must be provided at the system level.

12. Never use the product if the power cable is damaged. Check the power cable on a regular basis to ensure that it is in proper operating condition. By taking appropriate safety measures and carefully laying the power cable, ensure that the cable cannot be damaged and that no one can be hurt by e.g. tripping over the cable or suffering an electric shock.
13. The product may be operated only from TN/TT supply networks fused with max. 16 A (higher fuse only after consulting with the Rohde & Schwarz group of companies).
14. Do not insert the plug into sockets that are dusty or dirty. Insert the plug firmly and all the way into the socket. Otherwise, this can result in sparks, fire and/or injuries.
15. Do not overload any sockets, extension cords or connector strips; doing so can cause fire or electric shocks.
16. For measurements in circuits with voltages V_{rms} > 30 V, suitable measures (e.g. appropriate measuring equipment, fusing, current limiting, electrical separation, insulation) should be taken to avoid any hazards.
17. Ensure that the connections with information technology equipment comply with IEC 950/EN 60950.
18. Unless expressly permitted, never remove the cover or any part of the housing while the product is in operation. Doing so will expose circuits and components and can lead to injuries, fire or damage to the product.
19. If a product is to be permanently installed, the connection between the PE terminal on site and the product's PE conductor must be made first before any other connection is made. The product may be installed and connected only by a license electrician.
20. For permanently installed equipment without built-in fuses, circuit breakers or similar protective devices, the supply circuit must be fused in such a way that suitable protection is provided for users and products.
21. Do not insert any objects into the openings in the housing that are not designed for this purpose. Never pour any liquids onto or into the housing. This can cause short circuits inside the product and/or electric shocks, fire or injuries.
22. Use suitable overvoltage protection to ensure that no overvoltage (such as that caused by a thunderstorm) can reach the product. Otherwise the operating personnel will be endangered by electric shocks.
23. Rohde & Schwarz products are not protected against penetration of liquids, unless otherwise specified (see also safety instruction 1.). If this is not taken into account, there exists the danger of electric shock for the user or damage to the product, which can also lead to personal injury.
24. Never use the product under conditions in which condensation has formed or can form in or on the product, e.g. if the product was moved from a cold to a warm environment.
25. Do not close any slots or openings on the product, since they are necessary for ventilation and prevent the product from overheating. Do not place the product on soft surfaces such as sofas or rugs or inside a closed housing, unless this is well ventilated.
26. Do not place the product on heat-generating devices such as radiators or fan heaters. The temperature of the environment must not exceed the maximum temperature specified in the data sheet.
27. Batteries and storage batteries must not be exposed to high temperatures or fire. Keep batteries and storage batteries away from children. Do not short-circuit batteries and storage batteries.
    If batteries or storage batteries are improperly replaced, this can cause an explosion (warning: lithium cells).

Replace the battery or storage battery only with the matching Rohde & Schwarz type (see spare parts list). Batteries and storage batteries must be recycled and kept separate from residual waste. Batteries and storage batteries that contain lead, mercury or cadmium are hazardous waste. Observe the national regulations regarding waste disposal and recycling.

28. Please be aware that in the event of a fire, toxic substances (gases, liquids etc.) that may be hazardous to your health may escape from the product.
29. The product can be very heavy. Be careful when moving it to avoid back or other physical injuries.
30. Do not place the product on surfaces, vehicles, cabinets or tables that for reasons of weight or stability are unsuitable for this purpose. Always follow the manufacturer's installation instructions when installing the product and fastening it to objects or structures (e.g. walls and shelves).
31. Handles on the products are designed exclusively for personnel to hold or carry the product. It is therefore not permissible to use handles for fastening the product to or on means of transport such as cranes, fork lifts, wagons, etc. The user is responsible for securely fastening the products to or on the means of transport and for observing the safety regulations of the manufacturer of the means of transport. Noncompliance can result in personal injury or material damage.
32. If you use the product in a vehicle, it is the sole responsibility of the driver to drive the vehicle safely. Adequately secure the product in the vehicle to prevent injuries or other damage in the event of an accident. Never use the product in a moving vehicle if doing so could distract the driver of the vehicle. The driver is always responsible for the safety of the vehicle. The manufacturer assumes no responsibility for accidents or collisions.
33. If a laser product (e.g. a CD/DVD drive) is integrated in a Rohde & Schwarz product, do not use any other settings or functions than those described in the product documentation. Otherwise this may be hazardous to your health, since the laser beam can cause irreversible damage to your eyes. Never try to take such products apart, and never look into the laser beam.
34. Prior to cleaning, disconnect the product from the AC supply. Use a soft, non-linting cloth to clean the product. Never use chemical cleaning agents such as alcohol, acetone or diluent for cellulose lacquers.

Informaciones elementales de seguridad

¡Es imprescindible leer y observar las siguientes instrucciones e informaciones de seguridad!

El principio del grupo de empresas Rohde & Schwarz consiste en tener nuestros productos siempre al día con los estándares de seguridad y de ofrecer a nuestros clientes el máximo grado de seguridad. Nuestros productos y todos los equipos adicionales son siempre fabricados y examinados según las normas de seguridad vigentes. Nuestra sección de gestión de la seguridad de calidad controla constantemente que sean cumplidas estas normas. El presente producto ha sido fabricado y examinado según el comprobante de conformidad adjunto según las normas de la CE y ha salido de nuestra planta en estado impecable según los estándares técnicos de seguridad. Para poder preservar este estado y garantizar un funcionamiento libre de peligros, el usuario deberá atenerse a todas las indicaciones, informaciones de seguridad y notas de alerta. El grupo de empresas Rohde & Schwarz está siempre a su disposición en caso de que tengan preguntas referentes a estas informaciones de seguridad.

Además queda en la responsabilidad del usuario utilizar el producto en la forma debida. Este producto está destinado exclusivamente al uso en la industria y el laboratorio o, si ha sido expresamente autorizado, para aplicaciones de campo y de ninguna manera deberá ser utilizado de modo que alguna persona/cosa pueda sufrir daño. El uso del producto fuera de sus fines definidos o despreciando las informaciones de seguridad del fabricante queda en la responsabilidad del usuario. El fabricante no se hace en ninguna forma responsable de consecuencias a causa del mal uso del producto.

Se parte del uso correcto del producto para los fines definidos si el producto es utilizado dentro de las instrucciones de la correspondiente documentación de producto y dentro del margen de rendimiento definido (ver hoja de datos, documentación, informaciones de seguridad que siguen). El uso del producto hace necesarios conocimientos profundos y conocimientos básicas del idioma inglés. Por eso se debe tener en cuenta que el producto sólo pueda ser operado por personal especializado o personas minuciosamente instruidas con las capacidades correspondientes. Si fuera necesaria indumentaria de seguridad para el uso de productos de R&S, encontrará la información debida en la documentación del producto en el capítulo correspondiente. Guarde bien las informaciones de seguridad elementales, así como la documentación del producto y entréguela a usuarios posteriores.

Símbolos y definiciones de seguridad

⚠	18 kg	⚡	♨	⏚	⏚	⏚	⚠
Ver documentación de producto	Informaciones para maquinaria con un peso de > 18kg	Peligro de golpe de corriente	¡Advertencia! Superficie caliente	Conexión a conductor protector	Conexión a tierra	Conexión a masa conductora	¡Cuidado! Elementos de construcción con peligro de carga electroestática

I ○	⏻	⎓	∼	≂	⧈
Potencia EN MARCHA/PARADA	Indicación Stand-by	Corriente continua DC	Corriente alterna AC	Corriente continua/alterna DC/AC	El aparato está protegido en su totalidad por un aislamiento de doble refuerzo

Tener en cuenta las informaciones de seguridad sirve para tratar de evitar daños y peligros de toda clase. Es necesario de que se lean las siguientes informaciones de seguridad concienzudamente y se tengan en cuenta debidamente antes de la puesta en funcionamiento del producto. También deberán ser tenidas en cuenta las informaciones para la protección de personas que encontrarán en el capítulo correspondiente de la documentación de producto y que también son obligatorias de seguir. En las informaciones de seguridad actuales hemos juntado todos los objetos vendidos por el grupo de empresas Rohde & Schwarz bajo la denominación de „producto“, entre ellos también aparatos, instalaciones así como toda clase de accesorios.

Palabras de señal y su significado

PELIGRO Identifica un peligro directo con riesgo elevado de provocar muerte o lesiones de gravedad si no se toman las medidas oportunas.

ADVERTENCIA Identifica un posible peligro con riesgo medio de provocar muerte o lesiones (de gravedad) si no se toman las medidas oportunas.

ATENCIÓN Identifica un peligro con riesgo reducido de provocar lesiones de gravedad media o leve si no se toman las medidas oportunas.

AVISO Indica la posibilidad de utilizar mal el producto y a consecuencia dañarlo.

En la documentación del producto se emplea de forma sinónima el término CUIDADO.

Las palabras de señal corresponden a la definición habitual para aplicaciones civiles en el área económica europea. Pueden existir definiciones diferentes a esta definición en otras áreas económicas o en aplicaciones militares. Por eso se deberá tener en cuenta que las palabras de señal aquí descritas sean utilizadas siempre solamente en combinación con la correspondiente documentación de producto y solamente en combinación con el producto correspondiente. La utilización de las palabras de señal en combinación con productos o documentaciones que no les correspondan puede llevar a malinterpretaciones y tener por consecuencia daños en personas u objetos.

Informaciones de seguridad elementales

1. El producto solamente debe ser utilizado según lo indicado por el fabricante referente a la situación y posición de funcionamiento sin que se obstruya la ventilación. Si no se convino de otra manera, es para los productos R&S válido lo que sigue:
   como posición de funcionamiento se define por principio la posición con el suelo de la caja para abajo, modo de protección IP 2X, grado de suciedad 2, categoría de sobrecarga eléctrica 2, utilizar solamente en estancias interiores, utilización hasta 2000 m sobre el nivel del mar, transporte hasta 4.500 m sobre el nivel del mar.
   Se aplicará una tolerancia de ±10% sobre el voltaje nominal y de ±5% sobre la frecuencia nominal.
2. En todos los trabajos deberán ser tenidas en cuenta las normas locales de seguridad de trabajo y de prevención de accidentes. El producto solamente debe de ser abierto por personal especializado autorizado. Antes de efectuar trabajos en el producto o abrirlo deberá este ser desconectado de la corriente. El ajuste, el cambio de partes, la manutención y la reparación deberán ser solamente efectuadas por electricistas autorizados por R&S. Si se reponen partes con importancia para los aspectos de seguridad (por ejemplo el enchufe, los transformadores o los fusibles), solamente podrán ser sustituidos por partes originales. Después de cada recambio de partes elementales para la seguridad deberá ser efectuado un control de seguridad (control a primera vista, control de conductor protector, medición de resistencia de aislamiento, medición de la corriente conductora, control de funcionamiento).
3. Como en todo producto de fabricación industrial no puede ser excluido en general de que se produzcan al usarlo elementos que puedan generar alergias, los llamados elementos alergénicos (por ejemplo el níquel). Si se producieran en el trato con productos R&S reacciones alérgicas, como por ejemplo urticaria, estornudos frecuentes, irritación de la conjuntiva o dificultades al respirar, se deberá consultar inmediatamente a un médico para averiguar los motivos de estas reacciones.
4. Si productos / elementos de construcción son tratados fuera del funcionamiento definido de forma mecánica o térmica, pueden generarse elementos peligrosos (polvos de sustancia de metales pesados como por ejemplo plomo, berilio, níquel). La partición elemental del producto, como por ejemplo sucede en el tratamiento de materias residuales, debe de ser efectuada solamente por personal especializado para estos tratamientos. La partición elemental efectuada inadecuadamente puede generar daños para la salud. Se deben tener en cuenta las directivas nacionales referentes al tratamiento de materias residuales.
5. En el caso de que se produjeran agentes de peligro o combustibles en la aplicación del producto que debieran de ser transferidos a un tratamiento de materias residuales, como por ejemplo agentes refrigerantes que deben ser repuestos en periodos definidos, o aceites para motores, deberán ser tenidas en cuenta las prescripciones de seguridad del fabricante de estos agentes de peligro o combustibles y las regulaciones regionales para el tratamiento de materias residuales. Cuiden también de tener en cuenta en caso dado las prescripciones de seguridad especiales en la descripción del producto.

6. Ciertos productos, como por ejemplo las instalaciones de radiocomunicación RF, pueden a causa de su función natural, emitir una radiación electromagnética aumentada. En vista a la protección de la vida en desarrollo deberían ser protegidas personas embarazadas debidamente. También las personas con un bypass pueden correr peligro a causa de la radiación electromagnética. El empresario/usuario está comprometido a valorar y señalar áreas de trabajo en las que se corra un riesgo aumentado de exposición a radiaciones para evitar riesgos.
7. La utilización de los productos requiere instrucciones especiales y una alta concentración en el manejo. Debe de ponerse por seguro de que las personas que manejen los productos estén a la altura de los requerimientos necesarios referente a sus aptitudes físicas, psíquicas y emocionales, ya que de otra manera no se pueden excluir lesiones o daños de objetos. El empresario lleva la responsabilidad de seleccionar el personal usuario apto para el manejo de los productos.
8. Antes de la puesta en marcha del producto se deberá tener por seguro de que la tensión preseleccionada en el producto equivalga a la del la red de distribución. Si es necesario cambiar la preselección de la tensión también se deberán en caso dabo cambiar los fusibles correspondientes del producto.
9. Productos de la clase de seguridad I con alimentación móvil y enchufe individual de producto solamente deberán ser conectados para el funcionamiento a tomas de corriente de contacto de seguridad y con conductor protector conectado.
10. Queda prohibida toda clase de interrupción intencionada del conductor protector, tanto en la toma de corriente como en el mismo producto. Puede tener como consecuencia el peligro de golpe de corriente por el producto. Si se utilizaran cables o enchufes de extensión se deberá poner al seguro que es controlado su estado técnico de seguridad.
11. Si el producto no está equipado con un interruptor para desconectarlo de la red, se deberá considerar el enchufe del cable de distribución como interruptor. En estos casos deberá asegurar de que el enchufe sea de fácil acceso y nabejo (según la medida del cable de distribución, aproximadamente 2 m). Los interruptores de función o electrónicos no son aptos para el corte de la red eléctrica. Si los productos sin interruptor están integrados en bastidores o instalaciones, se deberá instalar el interruptor al nivel de la instalación.
12. No utilice nunca el producto si está dañado el cable eléctrico. Compruebe regularmente el correcto estado de los cables de conexión a red. Asegure a través de las medidas de protección y de instalación adecuadas de que el cable de eléctrico no pueda ser dañado o de que nadie pueda ser dañado por él, por ejemplo al tropezar o por un golpe de corriente.
13. Solamente está permitido el funcionamiento en redes de distribución TN/TT aseguradas con fusibles de como máximo 16 A (utilización de fusibles de mayor amperaje sólo previa consulta con el grupo de empresas Rohde & Schwarz).
14. Nunca conecte el enchufe en tomas de corriente sucias o llenas de polvo. Introduzca el enchufe por completo y fuertemente en la toma de corriente. Si no tiene en consideración estas indicaciones se arriesga a que se originen chispas, fuego y/o heridas.
15. No sobrecargue las tomas de corriente, los cables de extensión o los enchufes de extensión ya que esto pudiera causar fuego o golpes de corriente.

16. En las mediciones en circuitos de corriente con una tensión de entrada de Ueff > 30 V se deberá tomar las precauciones debidas para impedir cualquier peligro (por ejemplo medios de medición adecuados, seguros, limitación de tensión, corte protector, aislamiento etc.).
17. En caso de conexión con aparatos de la técnica informática se deberá tener en cuenta que estos cumplan los requisitos del estándar IEC950/EN60950.
18. A menos que esté permitido expresamente, no retire nunca la tapa ni componentes de la carcasa mientras el producto esté en servicio. Esto pone a descubierto los cables y componentes eléctricos y puede causar heridas, fuego o daños en el producto.
19. Si un producto es instalado fijamente en un lugar, se deberá primero conectar el conductor protector fijo con el conductor protector del aparato antes de hacer cualquier otra conexión. La instalación y la conexión deberán ser efectuadas por un electricista especializado.
20. En caso de que los productos que son instalados fijamente en un lugar sean sin protector implementado, autointerruptor o similares objetos de protección, el circuito de suministro de corriente deberá estar protegido de manera que usuarios y productos estén suficientemente protegidos.
21. Por favor, no introduzca ningún objeto que no esté destinado a ello en los orificios de la caja del aparato. No vierta nunca ninguna clase de líquidos sobre o en la caja. Esto puede producir cortocircuitos en el producto y/o puede causar golpes de corriente, fuego o heridas.
22. Asegúrese con la protección adecuada de que no pueda originarse en el producto una sobrecarga por ejemplo a causa de una tormenta. Si no se verá el personal que lo utilice expuesto al peligro de un golpe de corriente.
23. Los productos R&S no están protegidos contra líquidos si no es que exista otra indicación, ver también punto 1. Si no se tiene en cuenta esto se arriesga el peligro de golpe de corriente para el usuario o de daños en el producto lo cual también puede llevar al peligro de personas.
24. No utilice el producto bajo condiciones en las que pueda producirse y se hayan producido líquidos de condensación en o dentro del producto como por ejemplo cuando se desplaza el producto de un lugar frío a un lugar caliente.
25. Por favor no cierre ninguna ranura u orificio del producto, ya que estas son necesarias para la ventilación e impiden que el producto se caliente demasiado. No pongan el producto encima de materiales blandos como por ejemplo sofás o alfombras o dentro de una caja cerrada, si esta no está suficientemente ventilada.
26. No ponga el producto sobre aparatos que produzcan calor, como por ejemplo radiadores o calentadores. La temperatura ambiental no debe superar la temperatura máxima especificada en la hoja de datos.
27. Baterías y acumuladores no deben de ser expuestos a temperaturas altas o al fuego. Guardar baterías y acumuladores fuera del alcance de los niños. No cortocircuitar baterías ni acumuladores. Si las baterías o los acumuladores no son cambiados con la debida atención existirá peligro de explosión (atención células de litio). Cambiar las baterías o los acumuladores solamente por los del tipo R&S correspondiente (ver lista de piezas de recambio). Las baterías y acumuladores deben reutilizarse y no deben acceder a los vertederos. Las baterías y acumuladores que contienen plomo, mercurio o cadmio deben tratarse como residuos especiales. Respete en esta relación las normas nacionales de evacuación y reciclaje.
28. Por favor tengan en cuenta que en caso de un incendio pueden desprenderse del producto agentes venenosos (gases, líquidos etc.) que pueden generar daños a la salud.

29. El producto puede poseer un peso elevado. Muévalo con cuidado para evitar lesiones en la espalda u otras partes corporales.

30. No sitúe el producto encima de superficies, vehículos, estantes o mesas, que por sus características de peso o de estabilidad no sean aptas para él. Siga siempre las instrucciones de instalación del fabricante cuando instale y asegure el producto en objetos o estructuras (por ejemplo paredes y estantes).

31. Las asas instaladas en los productos sirven solamente de ayuda para el manejo que solamente está previsto para personas. Por eso no está permitido utilizar las asas para la sujeción en o sobre medios de transporte como por ejemplo grúas, carretillas elevadoras de horquilla, carros etc. El usuario es responsable de que los productos sean sujetados de forma segura a los medios de transporte y de que las prescripciones de seguridad del fabricante de los medios de transporte sean observadas. En caso de que no se tengan en cuenta pueden causarse daños en personas y objetos.

32. Si llega a utilizar el producto dentro de un vehículo, queda en la responsabilidad absoluta del conductor que conducir el vehículo de manera segura. Asegure el producto dentro del vehículo debidamente para evitar en caso de un accidente las lesiones u otra clase de daños. No utilice nunca el producto dentro de un vehículo en movimiento si esto pudiera distraer al conductor. Siempre queda en la responsabilidad absoluta del conductor la seguridad del vehículo. El fabricante no asumirá ninguna clase de responsabilidad por accidentes o colisiones.

33. Dado el caso de que esté integrado un producto de láser en un producto R&S (por ejemplo CD/DVD-ROM) no utilice otras instalaciones o funciones que las descritas en la documentación de producto. De otra manera pondrá en peligro su salud, ya que el rayo láser puede dañar irreversiblemente sus ojos. Nunca trate de descomponer estos productos. Nunca mire dentro del rayo láser.

34. Antes de proceder a la limpieza, desconecte el producto de la red. Realice la limpieza con un paño suave, que no se deshilache. No utilice de ninguna manera agentes limpiadores químicos como, por ejemplo, alcohol, acetona o nitrodiluyente.

Certified Quality System

DIN EN ISO 9001 : 2000
DIN EN 9100 : 2003
DIN EN ISO 14001 : 2004

DQS REG. NO 001954 QM UM

QUALITÄTSZERTIFIKAT

Sehr geehrter Kunde,
Sie haben sich für den Kauf eines Rohde & Schwarz-Produktes entschieden. Hiermit erhalten Sie ein nach modernsten Fertigungsmethoden hergestelltes Produkt. Es wurde nach den Regeln unseres Managementsystems entwickelt, gefertigt und geprüft. Das Rohde & Schwarz Managementsystem ist zertifiziert nach:

DIN EN ISO 9001:2000
DIN EN 9100:2003
DIN EN ISO 14001:2004

CERTIFICATE OF QUALITY

Dear Customer,
you have decided to buy a Rohde & Schwarz product. You are thus assured of receiving a product that is manufactured using the most modern methods available. This product was developed, manufactured and tested in compliance with our quality management system standards.
The Rohde & Schwarz quality management system is certified according to:

DIN EN ISO 9001:2000
DIN EN 9100:2003
DIN EN ISO 14001:2004

CERTIFICAT DE QUALITÉ

Cher Client,
vous avez choisi d'acheter un produit Rohde & Schwarz. Vous disposez donc d'un produit fabriqué d'après les méthodes les plus avancées. Le développement, la fabrication et les tests respectent nos normes de gestion qualité.
Le système de gestion qualité de Rohde & Schwarz a été homologué conformément aux normes:

DIN EN ISO 9001:2000
DIN EN 9100:2003
DIN EN ISO 14001:2004

ROHDE&SCHWARZ

Customer Support

Technical support – where and when you need it

For quick, expert help with any Rohde & Schwarz equipment, contact one of our Customer Support Centers. A team of highly qualified engineers provides telephone support and will work with you to find a solution to your query on any aspect of the operation, programming or applications of Rohde & Schwarz equipment.

Up-to-date information and upgrades

To keep your instrument up-to-date and to be informed about new application notes related to your instrument, please send an e-mail to the Customer Support Center stating your instrument and your wish.
We will take care that you will get the right information.

USA & Canada	Monday to Friday	(except US public holidays)
	8:00 AM – 8:00 PM	Eastern Standard Time (EST)
	Tel. from USA	888-test-rsa (888-837-8772) (opt 2)
	From outside USA	+1 410 910 7800 (opt 2)
	Fax	+1 410 910 7801
	E-mail	CustomerSupport@rohde-schwarz.com
East Asia	Monday to Friday	(except Singaporean public holidays)
	8:30 AM – 6:00 PM	Singapore Time (SGT)
	Tel.	+65 6 513 0488
	Fax	+65 6 846 1090
	E-mail	CustomerSupport@rohde-schwarz.com
Rest of the World	Monday to Friday	(except German public holidays)
	08:00 – 17:00	Central European Time (CET)
	Tel. from Europe	+49 (0) 180 512 42 42*
	From outside Europe	+49 89 4129 13776
	Fax	+49 (0) 89 41 29 637 78
	E-mail	CustomerSupport@rohde-schwarz.com

* 0.14 €/Min within the German fixed-line telephone network, varying prices for the mobile telephone network and in different countries.

ROHDE&SCHWARZ

Rohde&Schwarz Adressen

Firmensitz, Werke und Tochterunternehmen

Firmensitz

ROHDE&SCHWARZ GmbH & Co. KG
Mühldorfstraße 15 · D-81671 München
P.O.Box 80 14 69 · D-81614 München
Phone +49 (89) 41 29-0
Fax +49 (89) 41 29-121 64
info.rs@rohde-schwarz.com

Werke

ROHDE&SCHWARZ Messgerätebau GmbH
Riedbachstraße 58 · D-87700 Memmingen
P.O.Box 16 52 · D-87686 Memmingen
Phone +49 (83 31) 1 08-0
+49 (83 31) 1 08-1124
info.rsmb@rohde-schwarz.com

ROHDE&SCHWARZ GmbH & Co. KG
Werk Teisnach
Kaikenrieder Straße 27 · D-94244 Teisnach
P.O.Box 11 49 · D-94240 Teisnach
Phone +49 (99 23) 8 50-0
Fax +49 (99 23) 8 50-174
info.rsdts@rohde-schwarz.com

ROHDE&SCHWARZ závod
Vimperk, s.r.o.
Location Spidrova 49
CZ-38501 Vimperk
Phone +420 (388) 45 21 09
Fax +420 (388) 45 21 13

ROHDE&SCHWARZ GmbH & Co. KG
Dienstleistungszentrum Köln
Graf-Zeppelin-Straße 18 · D-51147 Köln
P.O.Box 98 02 60 · D-51130 Köln
Phone +49 (22 03) 49-0
Fax +49 (22 03) 49 51-229
info.rsdc@rohde-schwarz.com
service.rsdc@rohde-schwarz.com

Tochterunternehmen

R&S BICK Mobilfunk GmbH
Fritz-Hahne-Str. 7 · D-31848 Bad Münder
P.O.Box 20 02 · D-31844 Bad Münder
Phone +49 (50 42) 9 98-0
Fax +49 (50 42) 9 98-105
info.bick@rohde-schwarz.com

ROHDE&SCHWARZ FTK GmbH
Wendenschloßstraße 168, Haus 28
D-12557 Berlin
Phone +49 (30) 658 91-122
Fax +49 (30) 655 50-221
info.ftk@rohde-schwarz.com

ROHDE&SCHWARZ SIT GmbH
Am Studio 3
D-12489 Berlin
Phone +49 (30) 658 84-0
Fax +49 (30) 658 84-183
info.sit@rohde-schwarz.com

R&S Systems GmbH
Graf-Zeppelin-Straße 18
D-51147 Köln
Phone +49 (22 03) 49-5 23 25
Fax +49 (22 03) 49-5 23 36
info.rssys@rohde-schwarz.com

GEDIS GmbH
Sophienblatt 100
D-24114 Kiel
Phone +49 (431) 600 51-0
Fax +49 (431) 600 51-11
sales@gedis-online.de

HAMEG Instruments GmbH
Industriestraße 6
D-63533 Mainhausen
Phone +49 (61 82) 800-0
Fax +49 (61 82) 800-100
info@hameg.de

Weltweite Niederlassungen

Auf unserer Homepage finden Sie: www.rohde-schwarz.com

- Vertriebsadressen
- Serviceadressen
- Nationale Webseiten

Contents

Introduction

This manual describes all available programming interfaces.

The TSMW is a high-power platform for optimizing all conventional mobile radio networks. Two highly sensitive 20 MHz front ends for any input frequencies from 30 MHz to 6 GHz, a dual-channel preselector and an FPGA-based software-defined architecture offer unsurpassed performance while providing maximum flexibility and future-proofness. In addition to functioning as a mobile radio scanner, the TSMW makes also an ideal digital I/Q receiver for customer specific applications.

Related Documentation

- For further information about the TSMW and its functionality see the Operation Manual of R&S TSMW Radio Network Analyzer.
- For information of the general MATLAB functions see the online help in MATLAB.

R&S TSMW-K1

A special feature of the TSMW is its TSMW-K1 application (R&S TSMW-K1). The application provides flexible MATLAB interface as well as an equivalent C++ function interface for performing measurements directly on the TSMW and processing the results on the PC. This enables you, for example, not only to design and analyze receiver algorithms in MATLAB, but also to port them to C++ as a real time version. Or you can even perform technology-independent channel measurements, which can be used to simulate realistic fading scenarios in the lab.

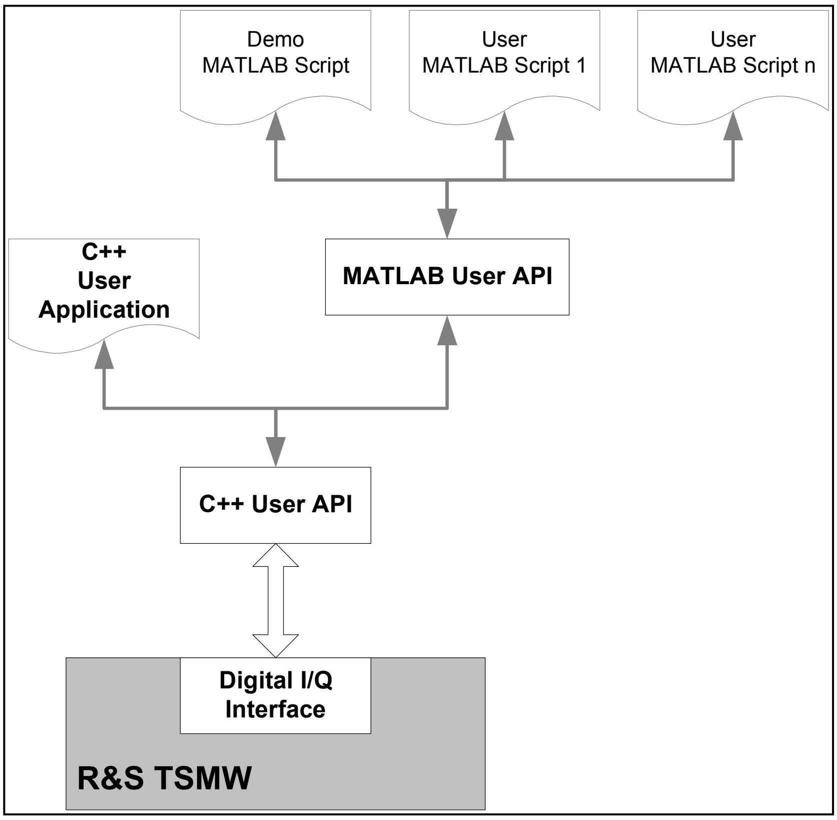


Figure 1: Digital I/Q interface overview

System Requirements

- 2GB RAM
- Gigabit Ethernet Adapter supporting up to 9kb Jumbo Frames
- CD-ROM drive (for installation only)
- Windows XP, Service Pack 2
- MATLAB® Software (including Filter Design Toolbox) Version R2007b
- TSMW
- Transfer rate hard disk: > 40 MB/sec (only for future IQ streaming extension)

Install

Installation of the TSMW-K1 software

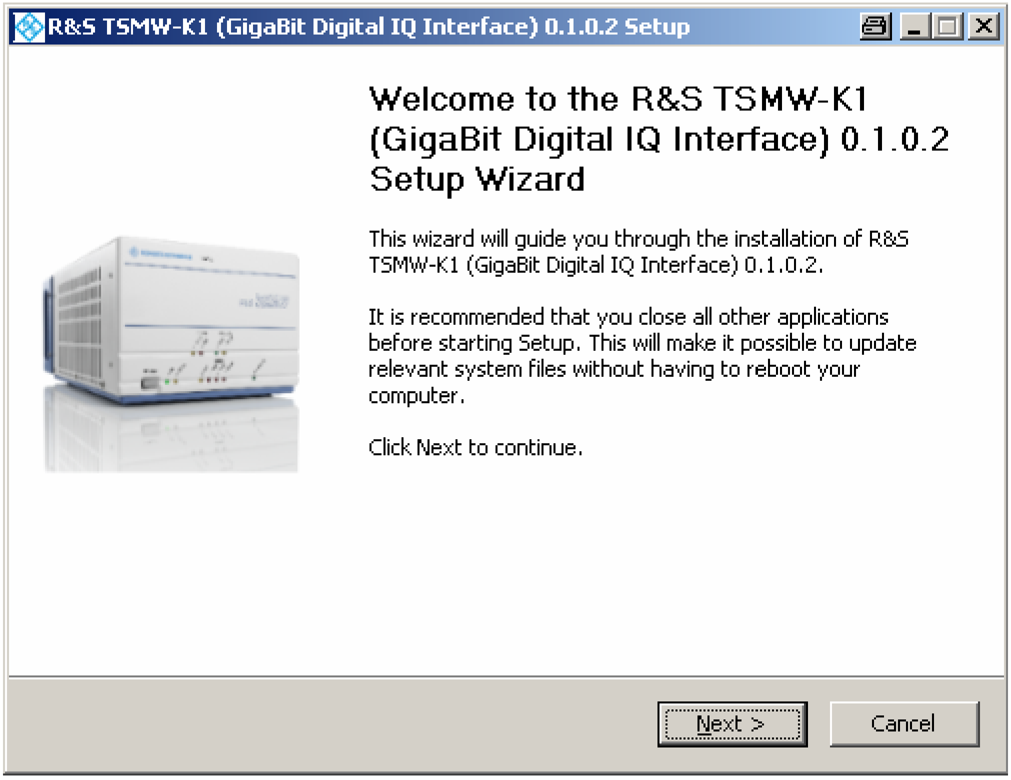
The installation of the software is controlled by Setup Wizard.
If you have no full version of Matlab R2007/R2008 installed, it is necessary to install the MATLAB Component Runtime (MCR). A compatible MCR version is on the CD-ROM.

INFO

Please keep in mind that the MCR allows only to run the compiled MATLAB Demo Application. Direct access to the corresponding TSMW MATLAB functions is only possible with a full MATLAB installation.

Procedure:

1. Insert the TSMW CD into the CD-ROM drive.
   The CD start menu opens automatically.
2. Select the menu item "Applications"-> "TSMW-K1 Setup".
   The "Setup Wizard" dialog starts immediately.



R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2 Setup

Welcome to the R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2 Setup Wizard

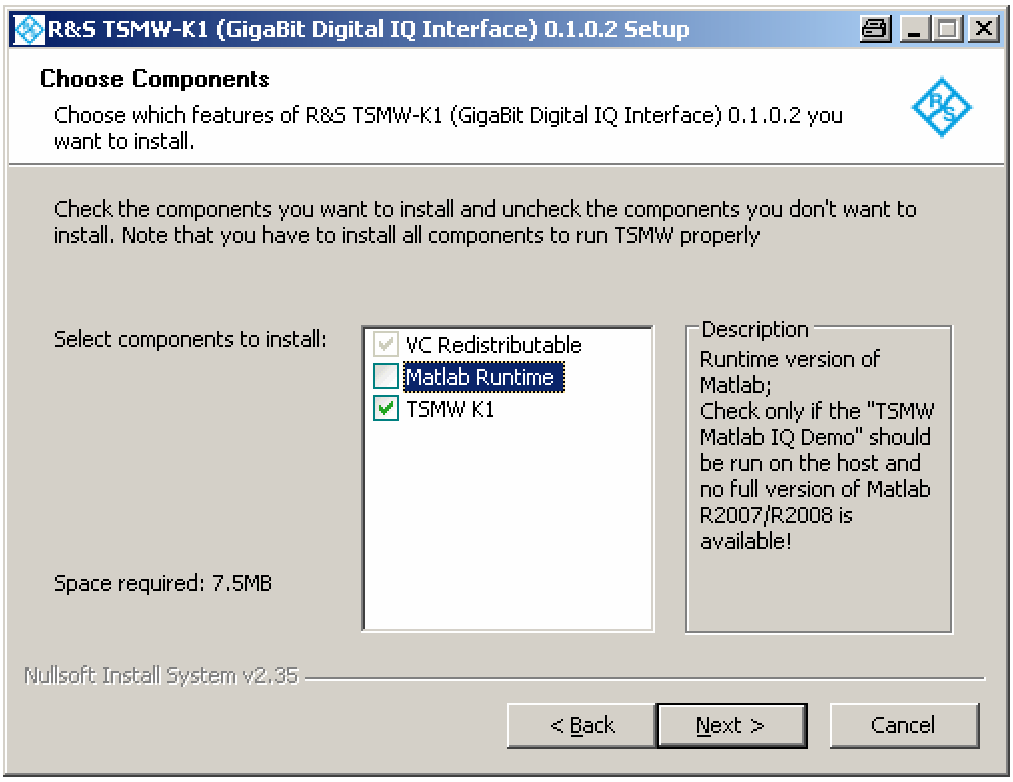
This wizard will guide you through the installation of R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Figure 2: Start dialog of the Setup Wizard

3. Click the “Next” button.
   The “License Agreement” window opens.
4. To continue the installation you have to agree with the End User License Agreement. Therefore click the “I Agree” button.
   The "Choose Components" window opens.



Choose Components

Choose which features of R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2 you want to install.

Check the components you want to install and uncheck the components you don't want to install. Note that you have to install all components to run TSMW properly

Select components to install:

☑ VC Redistributable
☐ Matlab Runtime
☑ TSMW K1

Description
Runtime version of Matlab;
Check only if the "TSMW Matlab IQ Demo" should be run on the host and no full version of Matlab R2007/R2008 is available!

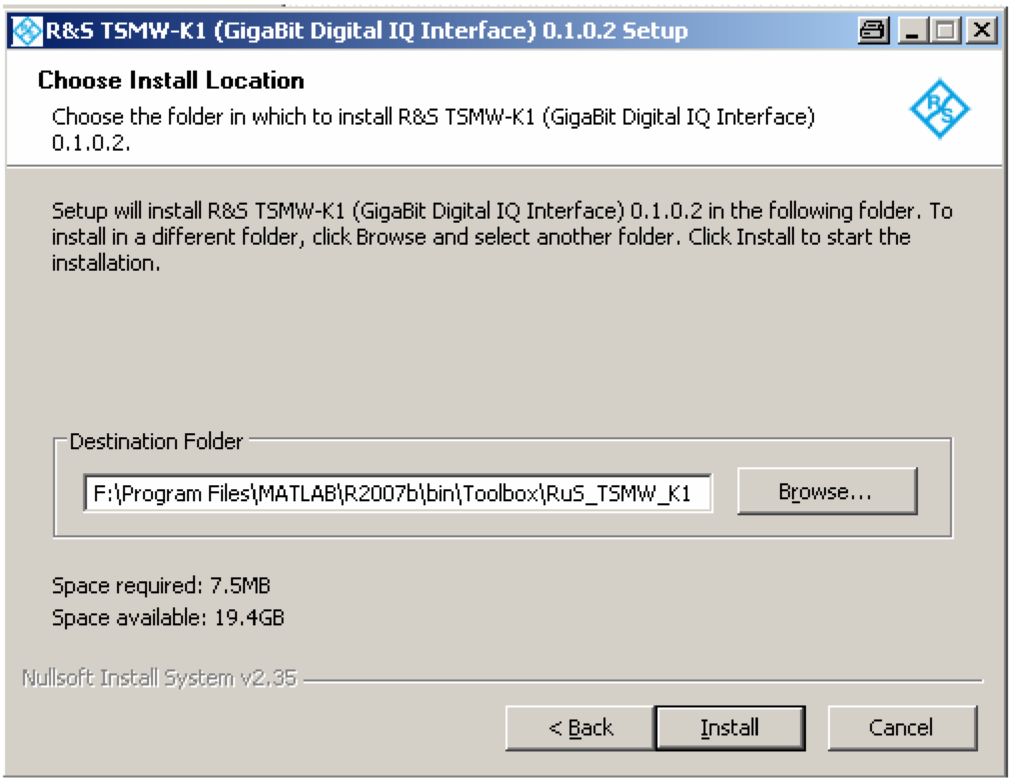
Space required: 7.5MB

Nullsoft Install System v2.35

Figure 3: Choose Component window

5. Check the desired components to install:
   a. Check the “TSMW-K1“ component to install the TSMW-K1 application (mandatory).
   b. Check the “Matlab Runtime” component if no full version of MATLAB R2007/R2008 is installed. (Optional)

6. Click the "Next" button to continue.
   The "Choose Install Location" window opens.



R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2 Setup

Choose Install Location

Choose the folder in which to install R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2.

Setup will install R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2 in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.

Destination Folder: F:\Program Files\MATLAB\R2007b\bin\Toolbox\RuS_TSMW_K1

Space required: 7.5MB
Space available: 19.4GB

Nullsoft Install System v2.35

Figure 4: Choose Install Location window

7. It is recommended to install all files to the MATLAB toolbox folder. The exact path depends on the installation of MATLAB.
   I. e.: C:\Program Files\MATLAB\R2007a\bin\toolbox\RuS_TSMW_K1
   Click the "Install" button to start the installation.
8. The installation of the TSMW-K1, VC Redistributable starts automatically.
9. During the installation a dialog window opens for the Redistributable Package. Follow the instruction and continue the installation.
10. If the component "Matlab Runtime" was selected the MATLAB Component Runtime (MCR) installer starts automatically.
    For detailed information how to install the MCR see Working with the MCR(http://www.mathworks.com/access/helpdesk/help/toolbox/compiler/f12-999353.html).

11. At the end of the installation process an installation complete window appears.
    Click the “Finish” button.
    The installation is finished.

    In the menu “Start”->”Programs”->”Rohde&Schwarz” exists a new menu folder for the TSMW-K1 software with all tools and information about it.

Set the path in MATLAB to the TSMW-K1 folder

It is required to add the installation path of the TSMW-K1 to the MATLAB environment in order to have access to the TSMW-K1 MATLAB functions.

Procedure:

1. Select in MATLAB on the menu bar "File" -> "Set Path…".
   The "Set Path" window opens.
2. Click the "Add Folder…" button.
   The "Browse For Folder" window opens.
3. Add the directory where the MATLAB wrapper functions are installed
   I.e.:
   `C:\Program Files\MATLAB\R2007a\bin\toolbox\RuS_TSMW_K1\Matlab`
4. Add the directory where the TSMW IQ Interface and related libraries are installed
   I.e.:
   `C:\Program Files\MATLAB\R2007a\bin\toolbox\RuS_TSMW_K1\lib`
5. Click the "Save" button.

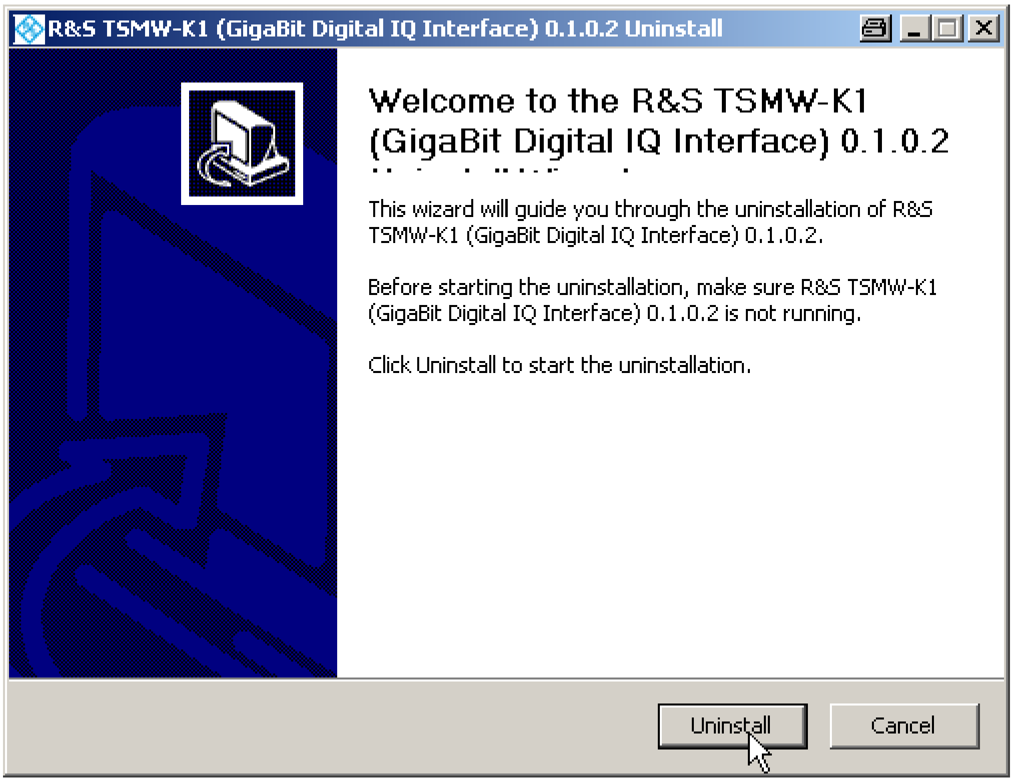
   The TSMW-K1 functions are now available in MATLAB.

Uninstall

The TSMW-K1 application offers a wizard to guide you through the uninstall process of the software.

Procedure:

1. Select
   "Start"->"Programs"->"Rohde&Schwarz"->"TSMW-K1 <ver. no.>"
   ->"Uninstall <ver. no.>".
   The uninstall wizard window opens.



R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2 Uninstall

Welcome to the R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2

This wizard will guide you through the uninstallation of R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2.

Before starting the uninstallation, make sure R&S TSMW-K1 (GigaBit Digital IQ Interface) 0.1.0.2 is not running.

Click Uninstall to start the uninstallation.

2. Follow the introduction of the wizard to uninstall the application.

   If no error occurs the application is successfully uninstalled.

TSMW signal processing block diagram

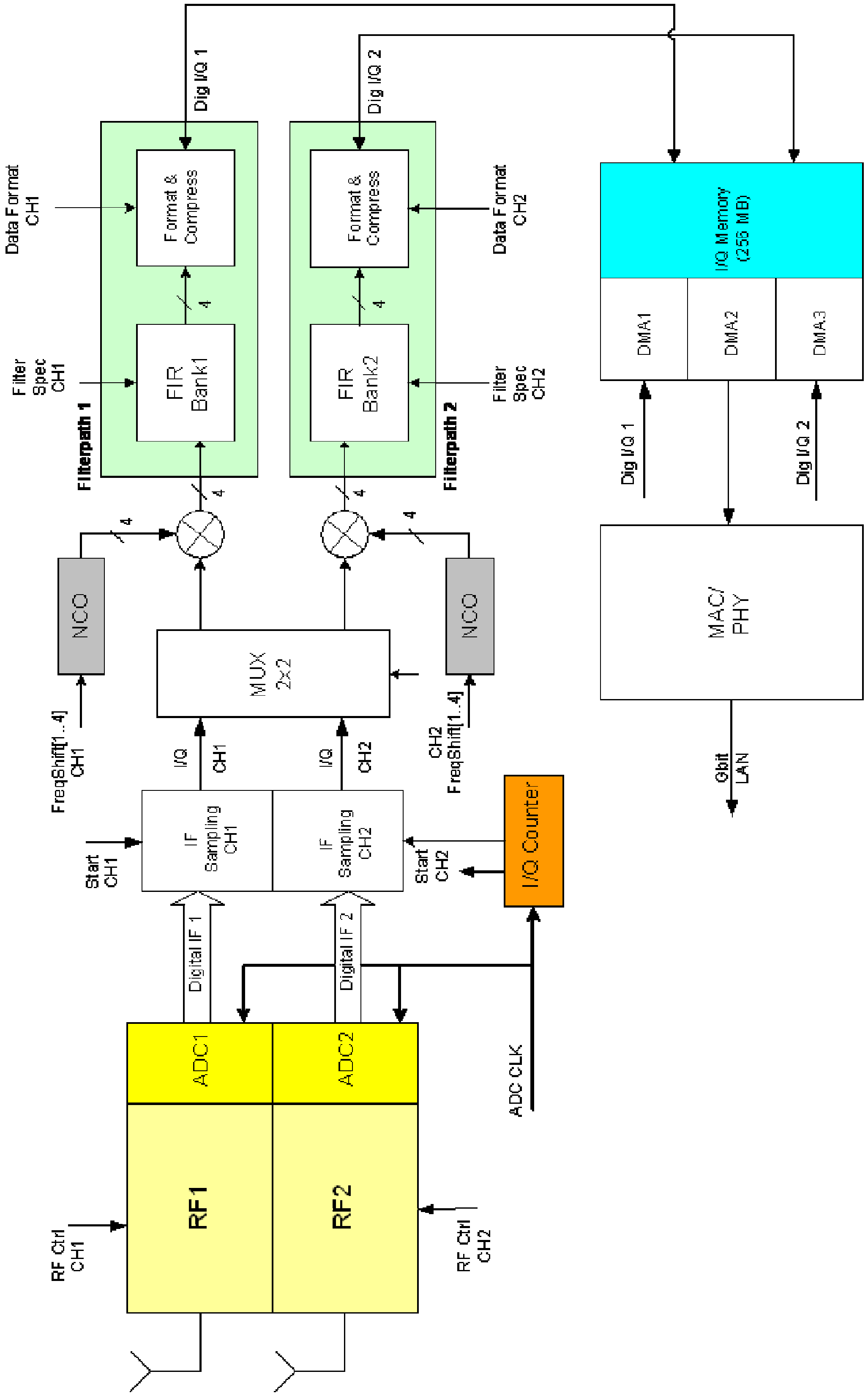


Figure 5: TSMW signal processing block diagramm

Function Overview

Function MATLAB (*.m file)	Function C++ (header files)	Description
`TSMWInitInterface`	`TSMWInitInterface_c`	Load and initialize the TSMW IQ interface library `TSMWIQInterface.dll`.
`TSMWReleaseInterface`	`TSMWReleaseInterface_c`	Unload the TSMW MATLAB interface library `TSMWIQInterface.dll` and disconnect from TSMW(s).
`TSMWConnect`	`TSMWConnect_c`	Establish a connection to the TSMW with the given IP address and given options.
`TSMWIQSetup`	`TSMWIQSetup_c`	Send a filter specification to the TSMW.
`TSMWGetLastError`	`TSMWGetLastError_c`	Return the error message of the occurred error.
`TSMWIQMeasure`	`TSMWIQMeasure_c`	Start a measurement with the given parameters.
`TSMWIQDataAvailable`	`TSMWIQDataAvailable_c`	Check how many measurement results from TSMW are available.
`TSMWGetIQResultParam`	`TSMWGetIQResultParam_c`	Get measurement result parameters.
`TSMWIQGetDataInt16`	`TSMWIQGetDataInt16_c`	Return the measurement data in 16 bit integer array format.
`TSMWIQGetDataInt32`	`TSMWIQGetDataInt32_c`	Return the measurement data in 32 bit integer array format.
`TSMWIQGetDataSingle`	`TSMWIQGetDataSingle_c`	Return the measurement data in single precision floating point array format
`TSMWIQGetDataDouble`	`TSMWIQGetDataDouble_c`	Return the measurement data in double precision floating point array format.
-	`TSMWIQGetDataSingleInterleaved_c`	Not yet implemented
-	`TSMWIQGetDataDoubleInterleaved_c`	Not yet implemented
`TSWMGetFIRParam`	-	Calculate FIR parameters for a given downsampling factor.
`TSMWShutdown`	`TSMWShutdown_c`	Shutdown the TSMW.
`TSMWTGEnable`	`TSMWTGEnable_c`	Enable the tracking generator (test) output.
`TSMWSetDAC`	`TSMWSetDAC_c`	Set the reference oscillator DAC value.
`TSMWGPSSync`	`TSMWGPSSync_c`	Enable or disable the GPS synchronization.
-	`TSMWGPSClearBuffer_c`	Clear the GPS NMEA buffer.

Function MATLAB (*.m file)	Function C++ (header files)	Description
`TSMWGetIQTime`	`TSMWGetIQTime_c`	Get current IQ time of given TSMW.
`TSMWTrigger`	`TSMWTrigger_c`	Set/get the TSMW trigger output/ input.
`TSMWGetVersion`	`TSMWGetVersion_c`	Get TSWW IQ Interface version.
`TSMWGetVersionText`	`TSMWGetVersionText_c`	Get TSMW IQ Interface version as text.
`TSMWGPSEnable`	`TSMWGPSEnable_c`	Enable or disable GPS data subscription from TSMW.
`TSMWGPSGetNMEALine`	`TSMWGPSGetNMEALine_c`	Get GPS NMEA text line.
`MeasCtrlTemplate`	-	Create a template for the MeasCtrl structure for double channel (MIMO) measurements with default values.
`MeasCtrlTemplate_RF1`	-	Create a template for the MeasCtrl structure for RF channel 1 measurements.
`MeasCtrlTemplate_RF2`	-	Create a template for the MeasCtrl structure for RF channel 2 measurements.

Table 1 Overview of the TSMW MATLAB and C++ IQ interface TSMW-K1 functions.

Function Description

All TSMW-K1 functions for the TSMW MATLAB IQ interface are explained in the following sub chapters. Additionally the corresponding C++ functions with the correct command syntax are mention. Most MATLAB parameters correspond to the C++ parameters.

INFO

Description of C++ functions

For more details about the available C++ functions and to get an overview of the input parameters and output value types, see the C++ header files. Inside the Appendix chapter C++ Header Files on p. 58 you will find an overview of the current C++ header files which was available during editorial deadline of this manual.
To see your current used C++ files go to `<TSMW-K1 installation directory>\inc` drive and look into the available C++ header files.

INFO

Example MATLAB scripts

Example MATLAB scripts are available in the example folder `<TSMW-K1 installation directory>\Examples\Matlab.`

TSMWGetVersion

Description:

The function returns the version number of the TSMW IQ interface.

Command syntax:

```
[VersionNo] = TSMWGetVersion;
```

Parameter(s):

-/-

Return value(s):

`VersionNo` Returns the version number in a 32-bit integer format. It represents the following 4-byte version code: MAJOR.MINOR.PATCH.QFE.

Corresponding C++ command:

```
int VersionNo = TSMWGetVersion_c();
```

TSMWGetVersionText

Description:

The function returns the version text of the TSMW IQ interface.

Command syntax:

```
[VersionText] = TSMWGetVersionText;
```

Parameter(s):

-/-

Return value(s):

`VersionText`	Returns the version text of the TSMW IQ Interface.

Corresponding C++ command:

```
char* VersionText = TSMWGetVersionText_c();
```

TSMWInitInterface

Description:

Load the TSMW IQ interface library `TSMWIQInterface.dll`.
The function `TSMWInitInterface` has to be called once before any other action can be performed on the TSMW.

Command syntax:

MATLAB

```
[ErrorText, ErrorCode] = TSMWInitInterface;
```

Parameter(s):

-/-

Return value(s):

`ErrorText`	If no error occurs an empty string is returned. Otherwise the message text of the error is returned.
`ErrorCode`	0 if successful. Otherwise the error code is returned.

Corresponding C++ command:

```
int ErrorCode = TSMWInitInterface_c();
```

TSMWReleaseInterface

Description:

Unload the external TSMW IQ interface library and disconnect from TSMW(s). Shall be called before the `dll` is unloaded.

INFO **MATLAB Interface Library**

To secure your MATLAB application from crash you have to call the function TSMWReleaseInterface before you can close the application.

Command syntax:

```
[ErrorCode] = TSMWReleaseInterface;
```

Parameter(s):

-/-

Return value(s):

`ErrorCode`	0 if successful. Otherwise the error code is returned.

Corresponding C++ command:

```
int ErrorCode = TSMWReleaseInterface_c();
```

TSMWConnect

Description:

Establish a connection to the TSMW with the given IP address and given options.

Command syntax:

```
[ErrorCode, TSMWID] = TSMWConnect(IPAddress,
                                        TSMWOptions)
```

Parameter(s):

`IPAddress`	IP address of the TSMW.
`TSMWOptions`	Specifies TSMW operating options. If omitted, default options will be used. That means both frontends and all amplifiers are enabled.

Return value(s):

`ErrorCode`	0 if successful. Otherwise the error code is returned.
`TSMWID`	The ID represents the TSMW connection.

Corresponding C++ command:

```
int ErrorCode = TSMWConnect_c(char* IPAddress,
                         TSMW_IQIF_MODE_t *pTSMWMode,
                         unsigned short *pTSMWID);
```

TSMWIQSetup

Description:

Transmits a filter specification to the TSMW. The FilterSpec.FilterNr gives the ID of the filter. Overrides any filter with the same ID.

Note

Resampling filter

In order to perform resampling on the TSMW, an appropriate resampling filter has to be transmitted to the TSMW. For filter design you might use the provided TSMW Filter Design Tool.

Command syntax:

```
[ErrorCode] = TSMWSetup(TSMWID, FilterSpec)
```

Parameter(s):

`TSMWID` Specifies the TSMW ID to use for measurement.

`FilterSpec` Filter specification structure.

For a detailed description of the structure refer to chapter Structure: FilterSpec on page 36 and to Filter Design GUI on p. 46.

Return value(s):

`ErrorCode` 0 if successful. Otherwise the error code is returned.

Corresponding C++ command:

```
[ErrorCode] = TSMWIQSetup_c(unsigned short TSMWID,
              TSMW_IQIF_FILTER_PARAM_t *pFilterParam,
                                      long *plCoeff);
```

TSMWGetLastError

Description:

Each function returns an error code. 0 if successful. Otherwise a number which defines the error code. To get the corresponding error text call the function TSMWGetLastError. The function returns the error information of the occurred error and reset the error code variable again to zero.

Note **ErrorCode variable**
Please keep in mind that the same ErrorCode variable is used for all functions. From this is follows that for any function the error code is written in this variable. Thus the error code before is always overwritten.

Command syntax:

```
[ErrorText, ErrorCode] = TSMWGetLastError
```

Parameter(s):

-/-

Return value(s):

`ErrorText`	Returns the error message of the last occurred error. If no error message exists an empty string is returned.
`ErrorCode`	If no error message exists the value zero is returned.

Corresponding C++ command:

```
char* ErrorText = TSMWGetLastError_c(
                                     int *pErrorCode);
```

TSMWIQMeasure

Description:

The function starts a new measurement with given parameters. The measurement parameters are defined in the structure `MeasCtrl`.

Command syntax:

```
[ErrorCode, MeasRequestID] = TSMWMeasure(TSMWID,
                                StartTimes,MeasData);
```

Parameter(s):

`TSMWID` Specifies the TSMW to use for measurement.

`StartTimes` Defines one or more start time(s) at which the measurement can be started. It is a multiple of 1/(395e6*18) seconds (approximately 45ns) and relative to startup of the TSMW.

If the parameter is empty, the measurement starts as soon as possible.

`MeasData` Measurement control structure. A template of the measurement control structure is provided by the MATLAB scripts “MeasCtrlTemplate”, “MeasCtrlTemplate_RF1” and “MeasCtrlTemplate_RF2”.

If the field “ChannelCtrl1” is present in the MeasCtrl structure, a measurement on RF-channel 1 is performed. If the field “ChannelCtrl2” is present, a measurement on RF-channel 2 is performed. A measurement on both channels is performed if both fields are present.

For further information about the fields refer to chapter Structure: MeasCtrl on page 33 and provided MATLAB example scripts.

Return value(s):

`ErrorCode` 0 if successful. Otherwise the error code is returned.

`MeasRequestID` ID for this measurement request.

Corresponding C++ command:

```
int ErrorCode =TSMWIQMeasure_c(
               unsigned short TSMWID,
               unsigned long *pMeasRequestID,
               unsigned __int64 *pStartTimes,
               long NoOfStartTimes,
               TSMW_IQIF_MEAS_CTRL_t *pMEAS_CTRL,
               TSMW_IQIF_CH_CTRL_t *pCHANNEL_CTRL1,
                                    TSMW_IQIF_CH_CTRL_t *pCHANNEL_CTRL2 );
```

TSMWIQDataAvailable

Description:

The function returns the number of the available measurement results. The value zero is returned when no data is available.

Command syntax:

```
[ErrorCode, NoOfDataBlocks] = TSMWIQDataAvailable();
```

Parameter(s):

-/-

Return value(s):

`ErrorCode`	0 if successful. Otherwise the error code is returned.
`NoOfDataBlocks`	Specifies the number of available data blocks.

Corresponding C++ command:

```
int ErrorCode = TSMWIQDataAvailable_c(
                                  long *pNoOfIQResults);
```

TSMWIQGetResultParam

Description:

This function returns the parameters of a measurement result. It will not delete the measurement result.

Command syntax:

```
[ErrorCode, IQResultParam] = TSMWIQDGetResultParam(
                             MeasRequestID, TimeOut);
```

Parameter(s):

`MeasRequestID` Specifies the measurement request ID of the measurement result to wait for. If 0, get parameters of next available measurement data block (with lowest measurement request ID).

`TimeOut` Defines the time in milliseconds to wait for the measurement result.

Return value(s):

`ErrorCode` 0 if successful. Otherwise the error code is returned.

`IQResultParam` Measurement result parameters.

For information about the fields refer to chapter Structure: TSMWIQResult on page 39.

Corresponding C++ command:

```
int ErrorCode = TSMWIQGetResultParam_c(
                          unsigned int MeasRequestID,
                          unsigned int TimeOut,
                          TSMW_IQ_RESULT *pIQResult);
```

TSMWIQGetDataInt16

Description:

This function unpacks the measurement result into 16 bit integer array format. The corresponding measurement result will be deleted.

Command syntax:

```
[ErrorCode,
IQResultParam,
IData,QData,Scaling,
Overflow,Calibrated] = TSMWIQGetDataInt16(TSMWID,
                          MeasRequestID,TimeOut,
                          NoOfSamples,NoOfChannels);
```

Parameter(s):

`TSMWID` Defines the TSMW.

`MeasRequestID` Specifies the measurement request ID of the measurement result to wait for. If 0, get parameters of next available measurement data block (with lowest measurement request ID).

`TimeOut` Defines the time in milliseconds to wait for the measurement response result.

`NoOfSamples` Defines the number of expected samples. This has to be equal to the number of samples specified in the measurement request (TSMWIQMeasure).

`NoOfChannels` Specifies the number of expected channels. This has to be equal to the total number of channels the corresponding measurement request covers.

I.e.: Measuring only on frontend 1 with only 1 sub-channel means `NoOfChannels` = 1. Measuring on frontend 1 and 2 with 1 sub-channel active on frontend 1 and 2 sub-channels on frontend 2 mean `NoOfChannels` = 3.

Return value(s):

`ErrorCode` 0 if successful. Otherwise the error code is returned.

`IQResultParam` Measurement result parameters.

For information about the fields refer to chapter Structure: TSMWIQResult on page 39.

`IData` In-phase data array. The first `NoOfSamples` data samples correspond to the first sub-channel of the first frontend. The next `NoOfSamples` data samples correspond to the next sub-channel (if applicable) of the first frontend etc..

`QData` Quadratur data array. Ordering as in IData.

`Scaling`	Reference level of the data in 1/100 dBm.
`Overflow`	Number of overflows that have occurred during the measurement.
`Calibrated`	0 if corresponding measurement setting is not calibrated. Otherwise 1 is returned.

Corresponding C++ command:

```
int ErrorCode = TSMWIQGetDataInt16_c(
                        unsigned short TSMWID,
                        unsigned int MeasRequestID,
                        unsigned int TimeOut,
                        TSMW_IQ_RESULT *pIQResult,
                        short* pReal,
                        short* pImag,
                        short* pScaling,
                        unsigned long* pOvfl,
                        unsigned int *pCalibrated,
                        unsigned int NoOfSamples,
                        unsigned int NoOfChannels,
                        int StartSampleDiv8,
                        int NoOfUnpckBitRes);
```

TSMWIQGetDataInt32

Description:

This function unpacks the measurement result into 32 bit integer array format. The corresponding measurement result will be deleted.
See also TSMWIQGetDataInt16.

Command syntax:

```
[ErrorCode,
IQResultParam,
IData,QData,Scaling,
Overflow,Calibrated] = TSMWIQGetDataInt32(TSMWID,
                           MeasRequestID,TimeOut,
                           NoOfSamples,NoOfChannels);
```

Parameter(s):

See TSMWIQGetDataInt16.

Return value(s):

See TSMWIQGetDataInt16.

Corresponding C++ command:

```
int ErrorCode = TSMWIQGetDataInt32_c(
                      unsigned short TSMWID,
                      unsigned int MeasRequestID,
                      unsigned int TimeOut,
                      TSMW_IQ_RESULT *pIQResult,
                      short* pReal,
                      short* pImag,
                      short* pScaling,
                      unsigned long* pOvfl,
                      unsigned int *pCalibrated,
                      unsigned int NoOfSamples,
                      unsigned int NoOfChannels,
                      int StartSampleDiv8,
                      int NoOfUnpckBitRes);
```

TSMWIQGetDataSingle

Description:

This function unpacks the measurement result into single precision floating point array format. The corresponding measurement result will be deleted.
See also TSMWIQGetDataInt16.

Command syntax:

```
[ErrorCode,
IQResultParam,
IData,QData,Scaling,
Overflow,Calibrated] = TSMWIQGetDataSingle(TSMWID,
                            MeasRequestID,TimeOut,
                            NoOfSamples,NoOfChannels);
```

Parameter(s):

See TSMWIQGetDataInt16.

Return value(s):

See TSMWIQGetDataInt16.

Corresponding C++ command:

```
int ErrorCode = TSMWIQGetDataSingle_c(
                        unsigned short TSMWID,
                       unsigned int MeasRequestID,
                       unsigned int TimeOut,
                       TSMW_IQ_RESULT *pIQResult,
                       short* pReal,
                       short* pImag,
                       short* pScaling,
                       unsigned long* pOvfl,
                       unsigned int *pCalibrated,
                       unsigned int NoOfSamples,
                       unsigned int NoOfChannels,
                       int StartSampleDiv8,
                       int NoOfUnpckBitRes);
```

TSMWIQGetDataDouble

Description:

This function unpacks the measurement result into double precision floating point array format. The corresponding measurement result will be deleted.
See also TSMWIQGetDataInt16.

Command syntax:

```
[ErrorCode,
IQResultParam,
IData,QData,Scaling,
Overflow,Calibrated] = TSMWIQGetDataDouble(TSMWID,
                           MeasRequestID,TimeOut,
                           NoOfSamples,NoOfChannels);
```

Parameter(s):

See TSMWIQGetDataInt16.

Return value(s):

See TSMWIQGetDataInt16.

Corresponding C++ command:

```
int ErrorCode = TSMWIQGetDataDouble_c(
                        unsigned short TSMWID,
                        unsigned int MeasRequestID,
                        unsigned int TimeOut,
                        TSMW_IQ_RESULT *pIQResult,
                        short* pReal,
                        short* pImag,
                        short* pScaling,
                        unsigned long* pOvfl,
                        unsigned int *pCalibrated,
                        unsigned int NoOfSamples,
                        unsigned int NoOfChannels,
                        int StartSampleDiv8,
                        int NoOfUnpckBitRes);
```

TSWMGetFIRParam

Description:

The function assists for custom filter design. It calculates the maximum number of FIR coefficients and an appropriate filter oversampling factor given a specific down sampling factor. The number of FIR filter tabs is given by NoOfCoeffs / CoeffOver.

Command syntax:

```
[NoOfCoeffs,OvsplFact] = TSWMGetFIRParam(Ndown,
                                         OvsplFact);
```

Parameter(s):

`Ndown`	Defines the fractional down sampling factor. Given the desired sampling rate fs, the down sampling factor is defined as fs/(395e6/18). I.e.: The desired sampling rate divided by the native TSMW sampling rate.
`OvsplFact`	If given, the specified oversampling factor is used. If not, an appropriate oversampling factor is used. It has to be 1, 2, 4, 8, 16 or 32. This parameter is optional.

Return value(s):

`NoOfCoeffs`	Maximum number of FIR filter coefficients. The number of FIR filter tabs is given by NoOfCoeffs / OvsplFact.
`OvsplFact`	Selected oversampling factor.

Corresponding C++ command:

`-/-`

TSMWShutdown

Description:

The function shuts the specified TSMW down.

Command syntax:

```
[ErrorCode] = TSMWShutdown(TSMWID);
```

Parameter(s):

`TSMWID`	Defines the TSMW to shut down.

Return value(s):

`ErrorCode`	0 if successful. Otherwise the error code is returned.

Corresponding C++ command:

```
int ErrorCode = TSMWShutdown_c(
                              unsigned short TSMWID);
```

TSMWTGEnable

Description:

The function enables the tracking (test) generator output.

Command syntax:

```
[ErrorCode] = TSMWTGEnable(TSMWID);
```

Parameter(s):

`TSMWID`	Defines the used TSMW.

Return value(s):

`ErrorCode`	0 if successful. Otherwise the error code is returned.

Corresponding C++ command:

```
int ErrorCode = TSMWTGEnable_c(unsigned short TSMWID,
                             unsigned int Frontends);
```

TSMWSetDAC

Description:

The function set the TSMW references oscillator Digital-to-Analog Converter (DAC) value for fine adjustment of the internal reference oscillator.

Note **GPS compatibility**

This function does only work when GPS synchronization is disabled.

Command syntax:

```
[ErrorCode] = TSMWSetDAC(TSMWID,Value);
```

Parameter(s):

`TSMWID`	Specifies the TSMW which shall be used.
`Value`	Specifies DAC value. Value range: 0..1023. Reasonable values are in the range 400..600.

Return value(s):

`ErrorCode`	0 if successful. Otherwise the error code is returned.

Corresponding C++ command:

```
int ErrorCode = TSMWSetDAC_c(unsigned short TSMWID,
                                        short Value);
```

TSMWGPSSync

Description:

The function enables or disables GPS synchronization. GPS synchronization is enabled by default.

Command syntax:

```
[ErrorCode] = TSMWGPSSync(TSMWID,Enable);
```

Parameter(s):

`TSMWID`	Specifies the TSMW ID.	
`Enable`	1	Enable GPS synchronization.
	0	Disable GPS synchronization.

Return value(s):

`ErrorCode` 0 if successful. Otherwise the error code is returned.

Corresponding C++ command:

```
int ErrorCode = TSMWGPSSync_c(unsigned short TSMWID,
                                          int Enable);
```

TSMWGPSEnable

Description:

The function enables or disables GPS data subscription from TSMW.

Command syntax:

```
[ErrorCode] = TSMWGPSEnable(TSMWID,Enable);
```

Parameter(s):

`TSMWID` Specifies the TSMW.

`Enable` 1: Enable GPS data subscription.
0: Disable GPS data subscription.

Return value(s):

`ErrorCode` 0 if successful. Otherwise the error code is returned.

Corresponding C++ command:

```
int ErrorCode = TSMWGPSEnable_c(
                          unsigned short TSMWID,
                          int NoOfBufferedLines,
                                          int Enable);
```

TSMWGPSGetNMEALine

Description:

The function gets the oldest available NMEA text in buffer.

Note **Configure NMEA buffer**

The number of buffered NMEA text lines can be parameterized by `TSMWGPSEnable_c`. The MATLAB wrapper function `TSMWGPSEnable` uses a fixed buffer size of 20.

Command syntax:

```
[NMEAText] = TTSMWGPSGetNMEALine();
```

Parameter(s):

-/-

Return value(s):

`NMEAText`	Oldest available NMEA text in buffer.

Corresponding C++ command:

```
char NMEAText = TSMWGPSGetNMEALine_c();
```

TSMWGetIQTime

Description:

The function retrieves an estimate of the current IQ index of the TSMW. The IQ index counter starts when a connection to the TSMW is established and counts the IQ-samples in the native sampling rate of 395/18 MS/s. It has a valid bit width of 48 bit.

Command syntax:

```
[ErrorCode, IQTime] = TSMWGetIQTime(TSMWID);
```

Parameter(s):

`TSMWID`	Defines the TSMW.

Return value(s):

`ErrorCode`	0 if successful. Otherwise the error code is returned.
`IQTime`	Specifies the current IQ index.

Corresponding C++ command:

```
int ErrorCode = TSMWGetIQTime_c(
                        unsigned short TSMWID,
                        unsigned long long *pIQTime);
```

TSMWTrigger

Description:

The function set and read TSMW trigger I/O data.

Command syntax:

```
[ErrorCode,
 TriggerIn1,
TriggerIn2]  = TSMWTrigger(TSMWID, TriggerLine, Out,
                           TriggerValue1, TriggerValue2);
```

Parameter(s):

`TSMWID` Specifies the TSMW ID to use for measurement.

`TriggerLine` Set/get data for different trigger lines:

1: For trigger line 1.

2: For trigger line 2.

3: For trigger line 1 and 2.

`Out` Set the corresponding trigger line:

0: Set to 'input'.

1: Set to 'output'

`TriggerValue1` If trigger line 1 is used as output
→ set to low if 0 and high if 1.

`TriggerValue2` If trigger line 2 is used as output
→ set to low if 0 and high if 1.

Return value(s):

`ErrorCode` 0 if successful. Otherwise the error code is returned.

`TriggerIn1` Current value of trigger line 1 if used as input
(0 = low, 1 = high).

`TriggerIn2` Current value of trigger line 2 if used as input
(0 = low, 1 = high).

Corresponding C++ command:

```
int ErrorCode = TSMWTrigger_c(unsigned short TSMWID,
                             int TriggerLine,
                             int Out,
                             int *pTrig1Value,
                             int *pTrig2Value);
```

MeasCtrlTemplate

Description:

The function creates a template for the `MeasCtrl` structure for double channel (MIMO) measurements with default values.

Command syntax:

```
[MeasCtrl] = MeasCtrlTemplate();
```

Parameter(s):

-/-

Return value(s):

`MeasCtrl`	Specifies a MeasCtrl structure with default values for double channel (MIMO) measurements. This means that the general measurement control sub-structure, the sub-structure for RF channel 1 as well as the sub-structure for RF channel 2 parameters will be set. For a detailed description of the structure refer to chapter Structure: MeasCtrl on p. 33.

Corresponding C++ command:

-/-

MeasCtrlTemplate_RF1

Description:

The function creates a template for the `MeasCtr`l structure for RF channel 1 measurements.

Command syntax:

```
[MeasCtrl] = MeasCtrlTemplate_RF1();
```

Parameter(s):

-/-

Return value(s):

`MeasCtrl` Specifies a `MeasCtrl` structure with default values for RF channel 1 measurements. This means that the general measurement control sub-structure and the sub-structure for RF channel 1 parameters will be set.

For a detailed description of the structure refer to chapter Structure: MeasCtrl on p. 33.

Corresponding C++ command:

-/-

MeasCtrlTemplate_RF2

Description:

The function creates a template for the `MeasCtrl` structure for RF channel 2 measurements.

Command syntax:

```
[MeasCtrl] = MeasCtrlTemplate_RF2();
```

Parameter(s):

-/-

Return value(s):

`MeasCtrl` Specifies a MeasCtrl structure with default values for RF channel 2 measurements. This means that the general measurement control sub-structure and the sub-structure for RF channel 2 parameters will be set.

For a detailed description of the structure refer to chapter Structure: MeasCtrl on p. 33.

Corresponding C++ command:

-/-

Structures

For easier working with the TSMW MATLAB IQ Interface API, several parameter structures are defined. The following sub chapters explain the structures with their fields.
Additionally the corresponding C++ structure is mention. Most C++ parameters correspond to the available MATLAB fields.

NOTE **Example of MATLAB structures**
Examples are available in the example folder
`<TSMW-K1 installation directory>\Examples\Matlab.`

Structure: MeasCtrl

Description:

The structure contains measurement control values. It is divisible into the following three parts:

- **General measurement control field**:
  Defines general measurement control values.
- **RF channel 1 control field**:
  Defines specific RF channel 1 control values.
- **RF channel 2 control field**:
  Defines specific RF channel 2 control values.

For easier handling in MATLAB these three components are put together in one structure `MeasCtrl`.
In C++ these measurement control structures are defined as two structure types (`typdef struct`):

- One structure for general measurement values:
  `struct TSMW_IQIF_MEAS_CTRL`
- One structure for independent RF channel control:
  `struct TSMW_IQIF_CH_CTRL`

MeasCtrl values for general measurement control

Description:

These fields allow to configure the general measurement control settings on the TSMW. For faultless use the values has to be set and submit over the Digital I/Q interface to the TSMW(s).

Field(s):

`MeasCtrl.NoOfSamples`	Specifies the number of IQ samples to measure.
`MeasCtrl.FilterType`	Specifies the used filter type: 0: Use predefined filters 1: Use user defined filters. (They have to be sent to the TSMW before).
`MeasCtrl.FilterNr`	If a predefined filter is used this field define the filter number.
`MeasCtrl.DataFormat`	Specifies I/Q data compression format for blockwise compression. 0: 2x 8 Bit per complex sample 1: 2x 12 Bit per complex sample 2: 2x 16 Bit per complex sample 3: 2x 20 Bit per complex sample
`MeasCtrl.AttStrategy`	Specifies the attenuation strategy. Currently unused. It is recommended to set to zero.
`MeasCtrl.Splitter`	Specifies the RF channel 1 splitter. Splitting the signal after preseletor to both frontends. 0: Disable splitter 1: Enable splitter
`MeasCtrl.Priority`	Specifies the priority of the measurement. Valid priority levels are 0 .. 16.

C++ structure:

```
typedef struct TSMW_IQIF_MEAS_CTRL
{
             unsigned long NoOfSamples;
             unsigned char FilterType;
             unsigned char FilterNr;
             unsigned char DataFormat;
             unsigned char AttStrategy;
             unsigned char Splitter;
             unsigned short Priority;
} TSMW_IQIF_MEAS_CTRL_t;
```

MeasCtrl values for RF channel 1 and RF channel 2

Description:

Each RF channel has its own fields. These fields contain measurement control values for the corresponding RF channel.

Field(s):

NOTE The following explanation shows only the values for the RF channel 1 control fields.
They correspond to the RF channel 2. Therefore replace the prefix "ChannelCtrl1"(RF channel 1) to "ChannelCtrl2" (RF channel 2).

`ChannelCtrl1.Frequency` Center frequency (Hz).

`ChannelCtrl1.UseOtherFrontend` Reserved for future use, has to be set to zero.

`ChannelCtrl1.NoOfChannels` Specifies the number of channels that shall be used (1..4). The received bandwidth of the selected frontend is split up into the corresponding number of channels.
I. e.: If 3 channels are used, the downsampling factor has to be at least 3.

`ChannelCtrl1.Attenuation` Specifies the attenuation to use (0..15dB).

Note:
Only the values 0,5,10 and 15 dB are calibrated.

`ChannelCtrl1.Preamp` Defines whether the preamp. shall be enabled or not.

0: Disable

1: Enable

`ChannelCtrl1.CalibInput` Defines whether the calibration input shall fit to the used frontend.

0: Disable

1: Enable

Note:
The calibration amplifier has to be enabled, too.

`ChannelCtrl1.FreqShift[]` Defines the frequency shift of each frequency shift from center frequency for each channel

`ChannelCtrl1.ChannelDelay[]` Specifies a separate delay in I/Q samples after filtering and downsampling for each channel.

`BlockSize` Reserved for future use, has to be zero.

`BlockSkip` Reserved for future use, has to be zero.

C++ structure:

```
typedef struct TSMW_IQIF_CH_CTRL
{
             unsigned __int64 Frequency;
             unsigned char UseOtherFrontend;
             unsigned char NoOfChannels;
             char Attenuation;
             unsigned char Preamp;
             unsigned char CalibInput;
             double FreqShift[4];
             long ChannelDelay[4];
             long BlockSize;
             long BlockSkip;
} TSMW_IQIF_CH_CTRL_t;
```

Structure: FilterSpec

Description:

The structure describes the filter parameter structure.

Field(s):

`FilterNr` Specifies the filter number.

`NoOfCoeffs` Specifies the number of filter coefficients.

`OversplFact` Specifies the filter coefficient oversampling factor.

`AvgFilterGaindB` Specifies the average filter gain in the pass band in dB. The result will be corrected by this value.

Note:
This value is the remaining filter gain after considering a factor of

	2^(-ResultShiftNumber).
`ResultShiftNumber`	Specifies the number of bits the result is shifted. Compensates for the filter gain.
`GroupDelay`	Specifies the Group delay of this filter in taps. A measurement will automatically be started “GroupDelay” samples before the given start time (if a start time was given).
`NDown`	Specifies the fractional down sampling factor. **Note:** Since the maximum number of filter coefficients and the oversampling factor depends on the downsampling factor, it is essential that these values match, for correct filtering/downsampling.

C++ structure:

```
typedef struct TSMW_IQIF_FILTER_PARAM
{
            unsigned short FilterNr;
            unsigned short NoOfCoeffs;
            unsigned char OvsplFact;
            double AvgFilterGaindB;
            unsigned char ResultShiftNumber;
            unsigned short GroupDelay;
            double Ndown;
} TSMW_IQIF_FILTER_PARAM_t;
```

Structure: TSMWOptions

Description:

The structure describes the Digital I/Q interface options for the TSMW.

Field(s):

`Frontends`

Specifies which frontend to enable

1: Enable frontend 1

2: Enable frontend 2

3: Enable both frontends

`AMPS_CH1`

Enable the frequency band for RF channel 1. Set to (2^32-1) for all bands (amplifiers).

`AMPS_CH2`

Enable the frequency band for RF channel 2. Set to (2^32-1) for all bands (amplifiers).

`Mode`

At the moment only the value zero is supported.

C++ structure:

```
typedef struct TSMW_IQIF_MODE
{
             unsigned char Frontends;
             unsigned long AMPS_CH1;
             unsigned long AMPS_CH2;
             unsigned char Mode;
}TSMW_IQIF_MODE_t;
```

Structure: TSMWIQResult

Description:

The structure describes the IQ result parameter structure.

Field(s):

`MeasRequestID`	Specifies the measurement request ID.
`StartTimeIQ`	Specifies the IQ-Counter value at which measurement was started. **Note:** Each individual channel will have a different start time, according to the specified `ChannelDelay` parameter in the measurement request structure.
`StartTimeHost`	Specifies the corresponding host time.
`Attenuation.1`	Specifies the current used attenuation in dB for each frontend 1.
`Attenuation.2`	Specifies the current used attenuation in dB for each frontend 2.
`Preamp.1`	Specifies for frontend 1: 0: Preamp was off 1: Preamp was on
`Preamp.2`	Specifies for frontend 2: 0: Preamp was off 1: Preamp was on

C++ structure:

```
typedef struct TSMW_IQ_RESULT
{
             unsigned long MeasRequestID;
             unsigned __int64 StartTimeIQ;
             unsigned __int64 StartTimeHost;
             unsigned char Attenuation[2];
             unsigned char Preamp[2];
} TSMW_IQ_RESULT_t;
```

Structure: TSMWIQIFStreamCtrl

Description:

The structure describes the IQ interface streaming control structure.

NOTE Streaming is currently not supported!

Field(s):

`StreamID`	Specifies the stream ID. Value range: 0..3

C++ structure:

```
typedef struct TSMW_IQIF_STREAM_CTRL
{
             unsigned char StreamID;
} TSMW_IQIF_STREAM_CTRL_t;
```

Demo Script

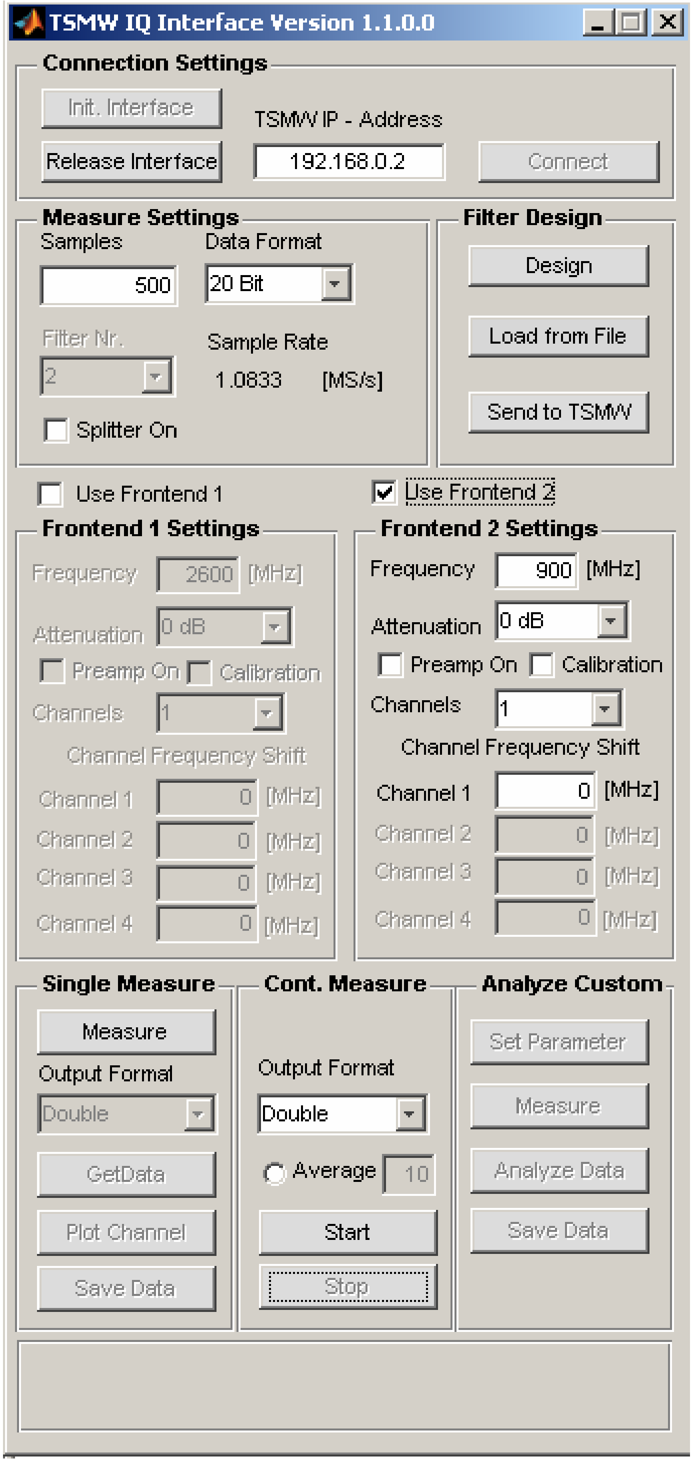
The TSMW MATLAB IQ Interface script is an example for using the TSMW-K1 API. On the basis of the example implementation the functionality of the TSMW-K1 API will be explained.

INFO **Source code of the TSMW MATLAB IQ Interface application**

The source code of the `TSMWIQInterfaceDemo.m` is available on the installed folder.

Main Window "TSMW Matlab IQ Interface"

The main window of the TSMW MATLAB IQ Interface script consists of different parts.
On the left you can control the connection to the TSMW and configure the measurement algorithm.
Every button is mapped to one or more TSMW-K1 MATLAB functions and shows an example implementation.
For viewing the measurement data a new window appears when a single measurement was done or a continuous measurement is performed (see Figure 7: Example plot from "Frontend 1 and 2".).



TSMW IQ Interface Version 1.1.0.0

Connection Settings

TSMW IP - Address: 192.168.0.2

Measure Settings

Samples: 500

Data Format: 20 Bit

Filter Nr.: 2

Sample Rate: 1.0833 [MS/s]

□ Splitter On

□ Use Frontend 1

☑ Use Frontend 2

Frontend 1 Settings

Frequency: 2600 [MHz]

Attenuation: 0 dB

□ Preamp On □ Calibration

Channels: 1

Channel Frequency Shift

Channel 1: 0 [MHz]

Channel 2: 0 [MHz]

Channel 3: 0 [MHz]

Channel 4: 0 [MHz]

Frontend 2 Settings

Frequency: 900 [MHz]

Attenuation: 0 dB

□ Preamp On □ Calibration

Channels: 1

Channel Frequency Shift

Channel 1: 0 [MHz]

Channel 2: 0 [MHz]

Channel 3: 0 [MHz]

Channel 4: 0 [MHz]

Single Measure

Output Format: Double

Cont. Measure

Output Format: Double

○ Average: 10

Figure 6:
Main window of TSMW IQ Interface demo application

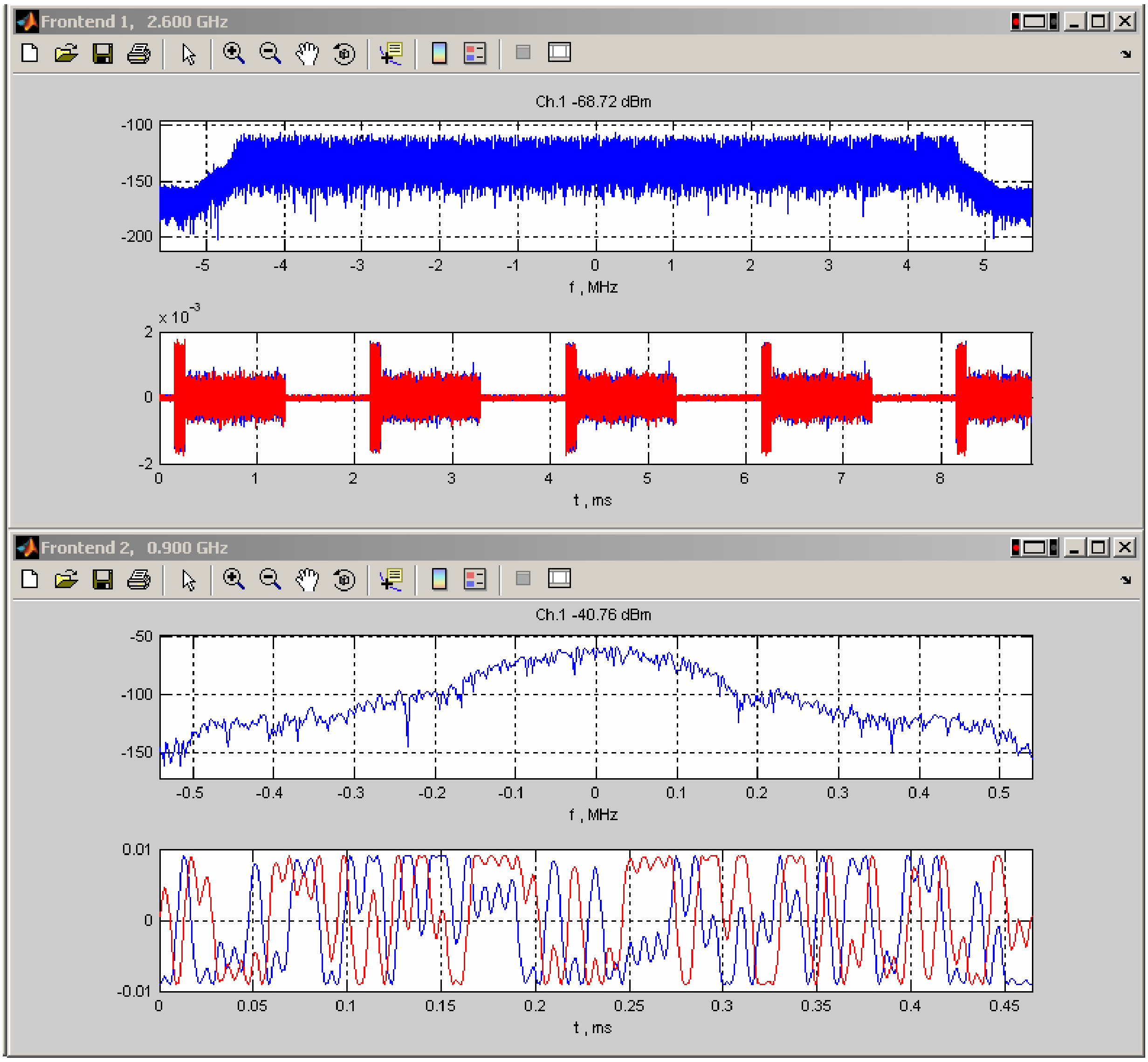


Figure 7: Example plot from “Frontend 1 and 2”.

The following tables list the functionality and the corresponding MATLAB functions of the TSMW MATLAB IQ Interface script.

Connection Settings

Name	Function/parameter/structure
Init.Interface	Function: TSMWInitInterface
ReleaseInterface	Function: TSMWReleaseInterface
TSMW IP-Address	Parameter: IPAddress
Connect	Function: TSMWConnect

Measure Settings

Name	Function/parameter/structure
Samples	Parameter: NoOfSamples
Data Format	Structure MeasCtrlTemplate Field: MeasCtrl.MeasCtrl.DataFormat
Splitter On	Structure MeasCtrlTemplate Field: MeasCtrl.MeasCtrl.Splitter
Filter Nr.	Structure MeasCtrlTemplate Field: MeasCtrl.MeasCtrl.FilterNr/FilterType

Filter Design

Name	Function/parameter/structure
Design	Calls an internal MATLAB function. For details see Filter Design GUI on page 46.
Load from file	Loads a saved filter specification.
Send to TSMW	Function: TSMWIQSetup.Transmits the new filter specification to the TSMW in order to make it available for measurements.

Frontend 1 Settings/ Frontend 2 Settings

Corresponds to MeasCtrl.ChannelCtrl1 / MeasCtrl.ChannelCtrl2.

Name	Function/parameter/structure
Use Frontend 1	If checked, a measurement will be performed on frontend 1. This means that only the field ChannelCtrl1 is present in the MeasCtrl structure
Use Frontend 2	If checked, a measurement will be performed on frontend 2. This means that only the field ChannelCtrl2 is present in the MeasCtrl structure .

Name	Function/parameter/structure
Frequency (in GHz)	Structure MeasCtrlTemplate Field: MeasCtrl.ChannelCtrl1/2.Frequency
Attenuation	Structure MeasCtrlTemplate

	Field: MeasCtrl.ChannelCtrl1/2.Attenuation
Preamp. On	Structure MeasCtrlTemplate Field: MeasCtrl.ChannelCtrl1/2.Preamp
Calibration	Structure MeasCtrlTemplate Field: MeasCtrl.ChannelCtrl1/2.CalibInput
Channnels	Structure MeasCtrlTemplate Field: MeasCtrl.ChannelCtrl1/2.NoOfChannels
Channels Frequency Shift 1 to 4 (in MHz)	Structure MeasCtrlTemplate Field: MeasCtrl.ChannelCtrl1/2.ChannelDelay1 to 4

Single Measure

Name	Function/parameter/structure
Measure	Function: TSMWIQMeasure
Output Format	Functions: TSMWIQGetDataInt16, TSMWIQGetDataInt32, TSMWIQGetDataSingle, TSMWIQGetDataDouble
GetData	Functions: TSMWIQDataAvailable, TSMWIQGetDataInt16, TSMWIQGetDataInt32,
Plot Channel	Shows results.
Save Data	

Cont. Measure

Name	Function/parameter/structure
Output Format	Functions: TSMWIQGetDataInt16, TSMWIQGetDataInt32, TSMWIQGetDataSingle, TSMWIQGetDataDouble
-/- Average (from number of measurements)	If the check box is activated then the average from number of measurements can be defined Parameter: NoOfIQResults
Start	Functions: TSMWIQMeasure, TSMWIQGetDataInt16, TSMWIQGetDataInt32 Parameter: StartTimes is empty
Stop	Functions: TSMWIQMeasure, TSMWIQGetDataInt16, TSMWIQGetDataInt32

Analyze Custom

For all GUI elements a corresponding dummy function in file `TSMWIQInterfaceDemo` exists.

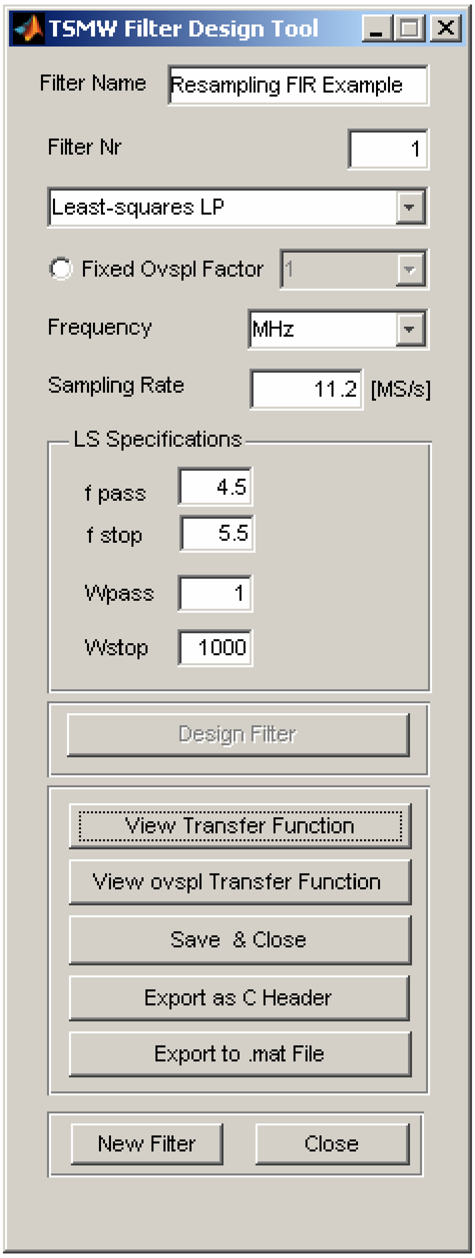
Name	Function/parameter/structure
Set Parameter	Function in file `TSMWIQInterfaceDemo.m` can be customized
Measure	Function in file `TSMWIQInterfaceDemo.m` can be customized
Analyze Data	Function in file `TSMWIQInterfaceDemo.m` can be customized

Save Data	Function in file `TSMWIQInterfaceDemo.m` can be customized

Filter Design GUI

The “TSMW Filter Design Tool” window creates and modifies filters for TSMW data measurements. The designed filters can be saved as `.mat` file for loading it back into the TSMW IQ Interface Demo Application and as C++ header file to include it into a C++ user application. The drop down menu offers several filter design algorithms with predefined values.
The filter designer base on the MATLAB Filter Design functions.

It is possible to check the new designed filter over the “View transfers function” functionality. The black vertical line in the "Filter Response" window corresponds to Sampling Rate / 2, i.e. the maximum frequency without aliasing.



Filter Name: Resampling FIR Example

Filter Nr: 1

Least-squares LP

○ Fixed Ovspl Factor: 1

Frequency: MHz

Sampling Rate: 11.2 [MS/s]

LS Specifications

f pass: 4.5

f stop: 5.5

Wpass: 1

Wstop: 1000

Figure 8: Example view "TSMW Filter Design Tool"

The following table describes the functionality and the corresponding MATLAB functions of the “TSMW Filter Design Tool” window.

Name	Function/parameter/structure
Filter Name	Structure FilterSpec Field: FilterSpec.Name
Filter Nr.	Structure FilterSpec Field: FilterSpec.FilterNr
-/-	Selected the desired filter type.
Fixed Ovspl Factor	If the box is checked an over sampling factor can be set. Structure FilterSpec Field: FilterSpec.OvsplFact

Name	Function/parameter/structure
Design Filter	
View Transfer Function	The function checks the designed filter.
View ovrspl Transfer ...	
Save & Close	Save the designed filter and return to the main window. The designed filter is available for measure purpose inside the MATLAB Demo script.
Export as C Header	
Export to .mat File	Export the designed filter to an M-file.
New Filter	Reset the values and start to design a new filter.
Close	Close the "TSMW Filter Design Tool” window without saving and return to the main window.

How to ...

This chapter describes typical use cases which can perform with the TSMW MATLAB IQ Interface demo application.

Use the compiled TSMW MATLAB IQ Interface application

Prerequisite:

- The TSMW-K1 software is installed on the PC.
  For detailed information how to install refer to chapter Install on page 2.
- The MCR environment is installed.
- The TSMW is successful connected to the PC.

Procedure:

1. Select "Start"->"Programs"->"Rohde&Schwarz"->"TSMW-K1 <ver. no.>" -> "TSMW MatlabDemo IQ".

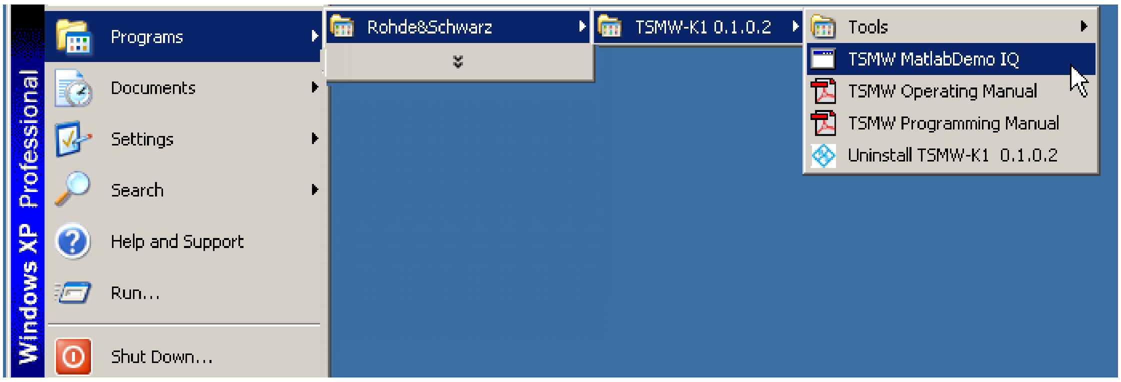


Figure 9: Start demo application

The main window of the TSMW MATLAB IQ Interface application opens.

Use the TSMW Matlab IQ Interface Application (not compiled)

Procedure:

1. Start MATLAB.
2. On the command window type :
   “TSMWIQInterfaceDemo”

The TSMW Matlab IQ Interface application starts and the main window of the application open.

Use native C++ Interface

- Copy all files from
`C:\Program Files\MATLAB\R2007a\bin\toolbox\RS_TSMWIQInterface\lib`
including subfolders to your C++ user application folder.

Use user specific MATLAB script

… find under <TSMW K1 install dir>\Examples\Matlab.

Connect to the TSMW

Prerequisite:

- A LAN connection between the PC and the TSMW is required.
How to link the software to TSMW refer to the corresponding TSMW operation manual.

Procedure:

1. Start the TSMW MATLAB IQ Interface application.
The TSMW MATLAB IQ Interface script window opens.
2. Click on the "Init. Interface" button.
An entry field for the TSMW IP address appears.
3. Enter the IP address of the TSMW device.
4. Click the "Connect" button.

A connection to the TSMW is established.

Create a filter

Prerequisites:

- The TSMW MATLAB IQ Interface application is running.
- A connection to a TSMW is established.

Procedure:

1. On the “TSMW MATLAB IQ Interface” window click the "Design" button.
   The Filter Design GUI opens.
   Enter the desired filter values.
   A new filter is created.
2. Click “View Transfer Function” to check the designed filter.
3. Save the filter: It exist 3 kinds to save the filter and leave the window:
   a. Click the “Save&Close” button. The filter is available for the current instance of TSMW MATLAB IQ Interface application and the user returns to the main window.
   b. Click the “Export to .mat File” button. The filter is saved in mat File format and available for later use.
   Click the “Close” button to return to the main window.
   c. Click the "Export to C Header" button. The filter is saved as C++ Header.

A new filter is created.

NOTE **Submit filter**
To make a filter chooseable you have to submit the filter explicitly to TSMW.
How to submit the filter see Submit filter to TSMW on p. 51.

Submit filter to TSMW

Prerequisites:

- The Demo GUI application is running.
- A connection to the TSMW is established.

Procedure:

1. Click the “Load from File” to load a filter or design a new filter over the “Design” button (see procedure Create a filter).
2. Click the "Send to TSMW" button.

The filter is transferred to the TSMW.
Now the filter is available in “Frontend 1” and ”Frontend 2” over the “Filter Nr.” drop down menu for further use.

Modify the measurement values

Prerequisites:

- The TSMW MATLAB IQ Interface application is running and a connection to the TSMW is established.

Procedure:

1. Edit the general values in the group box "Measure Settings".
   Please refer to chapter Structure: MeasCtrl on page 33 for an explanation of the parameters.
2. Check one check box "Use Frontend 1" or "Use Frontend 2" to activate the corresponding group box "Frontend 1 Settings" or "Frontend 2 Settings".
3. Edit the values in the activated group box "Frontend 1 Settings" or "Frontend 2 Settings".
   Please refer to chapter Structure: MeasCtrl on page 33 for an explanation of the parameters.
   a. Before a filter can selected in the "Filter Nr." drop down menu, it has to be designed.
      How to design a filter and make it available are explained in procedure Create a filter and Submit filter to TSMW on page 51.

The measurement settings are modified.
Now a measurement can be performed.

Perform a continuous measurement

Prerequisites:

- The TSMW MATLAB IQ Interface application is running and a connection to the TSMW is established.
- The measurement settings are defined.

Procedure:

1. It is possible to customize the average result of a number of measurements.
   a) Therefore click on the "Average Result" check box.
      The "No. of. Meas." entry field will be activated.
   b) Enter the desired number of measurements for the generation of average results.
2. Select from the drop down list box "Output Format" the desired output width.
3. Click the "Start" button to start the continuous measurement.
   The measurement starts and the data are available on the application.
4. Click the "Stop" button to stop the measurement.

A measurement was performed.

Perform a single measurement

Prerequisites:

- The TSMW MATLAB IQ Interface application is running and a connection to the TSMW is established.
- The measurement settings are defined.

Procedure:

1. Click the "Measure" button.
   The TSMW starts a measurement. Until now no measurement data are transferred to the MATLAB application.
2. Select from the drop down list box "Output Format" the desired output width.
3. Click the "GetData" button to transfer the measurement result to the application.
4. Click the “Plot Channel” button to visualize the measurement.

A measurement was performed.

Release the connection to the TSMW IQ interface

- The TSMW MATLAB IQ Interface script is running and a connection to the TSMW is established.
- All measurements are stopped.

Procedure:

1. Click the "ReleaseInterface" button.

The connection to the TSMW is released.

Appendix

TSMW-K1 (R&S® TSMW-K1)

C++ Header Files

TSMWIQInterfaceTypes.h

```
/**
 * TSMW Project   TSMW IQ Interface DLL
 *
 * @file          TSMWIQInterfaceTypes.h
 * @abstract
 *
 * @copyright     (c) 2007 Rohde & Schwarz GmbH & Co. KG, Munich
 * @author        Markus Herdin, Johannes Dommel
 * @version
 *    06.11.2007  Hd start
 *
 */

#ifndef  _TSMWIQINTERFACETYPES_H
#define  _TSMWIQINTERFACETYPES_H

#define IQSAMPLINERATE_HZ    (395e6 / 18)

// TSMW IQ interface mode structure
typedef struct TSMW_IQIF_MODE
{
   unsigned char    Frontends;             // Specifies which frontend to
enable
                                           //   1: Enable frontend 1
                                           //   2: enable frontend 2,
                                           //   3: enable both frontends
   unsigned long    AMPS_CH1;              // A combination of bits that
define which preselector bands (amplifier)
                                           // to activate for rf channel 1.
Each bit corresponds to a specific frequency band
                                           //   Bit 0: <0.6GHz
                                           //   Bit 1: 0.6 - 1.2GHz
                                           //   Bit 2: 1.2 - 1.7GHz
                                           //   Bit 3: 1.7 - 2.5GHz
                                           //   Bit 4: >2.5GHz
   unsigned long    AMPS_CH2;              // Same for rf channel 2
   unsigned char    Mode;                  // 0: Standard mode, 1:
Calibration mode (shall not be used)
} TSMW_IQIF_MODE_t;



// Filter parameter structure
typedef struct TSMW_IQIF_FILTER_PARAM
{
   unsigned short   FilterNr;              // Number of filter
   unsigned short   NoOfCoeffs;            // Number of filter coefficients
   unsigned char    OvsplFact;             // Filter coefficient
oversampling factor
   double           AvgFilterGaindB;       // Average filter gain in the
passband in dB. The result will be corrected by this value.
                                           // NOTE: This value is the
remaining filter gain after considering a factor
                                           // of 2^(-ResultShiftNumber)
   unsigned char    ResultShiftNumber;     // No of bits the result is
shifted. Compensates for the filter gain.
   unsigned short   GroupDelay;            // Group delay of this filter in
taps
   double           Ndown;                 // Downsampling factor this
filter was designed for.
} TSMW_IQIF_FILTER_PARAM_t;


// Measurement control structure
typedef struct TSMW_IQIF_MEAS_CTRL
{

   unsigned long        NoOfSamples;       // Number of IQ samples to
measure
   unsigned char        FilterType;        // Filter type:
                                           //   0: Use predefined filters
                                           //   1: Use userdefined filters
(they have to be sent to the TSMW, beforehand)
   unsigned char        FilterNr;          // Number of the filter that
shall be used
   unsigned char        DataFormat;        // IQ-data compression format for
blockwise compression
```

```
                                            //   0: 2 x  8 Bit per complex
sample
                                            //   1: 2 x 12 Bit
                                            //   2: 2 x 16 Bit
                                            //   3: 2 x 20 Bit
   unsigned char        AttStrategy;        // Attenuation strategy,
currently unused, shall be set to zero
   unsigned char        Splitter;           // RF channel 1 splitter to split
signal after preselector to
                                            // both frontends.
                                            //   0: Disable splitter
                                            //   1: Enable splitter
   unsigned short       Priority;           // Relative priority of this
meas.req., Valid range: 0 .. 15,
                                            // 0 is lowest priority, 15
highest
} TSMW_IQIF_MEAS_CTRL_t;


// Channel control structure
typedef struct TSMW_IQIF_CH_CTRL
{
   unsigned __int64   Frequency;            // Center frequency in Hz
   unsigned char      UseOtherFrontend;     // Defines which frontend shall
be used by this filter channel
                                            //   0: Use frontend 1 in case of
filter channel 1 and 2 in case of filter channel 2
                                            //   0: Use frontend 2 in case of
filter channel 1 and 1 in case of filter channel 2
   unsigned char      NoOfChannels;         // Number of channels that shall
be used (1..4). This means the receive
                                            // bandwidth of the selected
frontend is split up into the corresponding
                                            // number of channels.
                                            // NOTE: The downsampling factor
has to be larger or equal to the
                                            // number of channels. I.e. if 3
channels are used, the downsampling
                                            // factor has to be at least 3.
   char               Attenuation;          // Attenuation to use (0..15dB)
                                            // NOTE: Only the values 0, 5, 10
and 15dB are calibrated
   unsigned char      Preamp;               // Defines whether the preamp
shall be enabled or not
                                            //   0: Disable
```

```
                                           //   1: Enable
   unsigned char      CalibInput;          // Defines whether the
calibration input shall fed to the used frontend
                                           //   0: Disable
                                           //   1: Enable
                                           // NOTE: The calibration
amplifier has to be enabled, too (see TSMW_IQIF_MODE)

   double             FreqShift[4];        // Frequency shift from center
frequency in Hz for each subchannel
   long               ChannelDelay[4];     // Individual delay in taps for
each subchannel (after filtering/resampling)

   long               BlockSize;           // Block size, has to be larger
than 2
   long               BlockSkip;           // Samples to skip between blocks
} TSMW_IQIF_CH_CTRL_t;

// IQ result parameter structure
typedef struct TSMW_IQ_RESULT
{
   unsigned long      MeasRequestID;       // Meas.request ID
   unsigned __int64   StartTimeIQ;         // IQ-Counter value at which
measurement was started.
                                           // NOTE: Each individual channel
will have a different start time,
                                           // according to the specified
"ChannelDelay" parameter in the
                                           // meas.request structure.

   unsigned __int64   StartTimeHost;       // Corresponding host time

   unsigned char      Attenuation[2];      // Actually used attenation in dB
for each frontend
   unsigned char      Preamp[2];           // For each frontend: 0: Preamp
was off, 1: Preamp was on
} TSMW_IQ_RESULT_t;

// Streaming control structure
typedef struct TSMW_IQIF_STREAM_CTRL
{
   unsigned char        StreamID;          // Stream ID, valid range: 0..3
```

```
} TSMW_IQIF_STREAM_CTRL_t;

#endif // _TSMWIQINTERFACETYPES_H
```

TSMWIQInterfaceFunc.h

```
/**
 * TSMW Project   TSMW IQ Interface DLL
 *
 * @file          TSMWIQInterfaceFunc.h
 * @abstract
 *
 * @copyright     (c) 2007 Rohde & Schwarz GmbH & Co. KG, Munich
 * @author        Markus Herdin, Johannes Dommel
 * @version
 *    06.11.2007  Hd start
 *
 */

#include "TSMWIQInterfaceTypes.h"

#ifndef  _TSMWIQINTERFACEFUNC_H
#define  _TSMWIQINTERFACEFUNC_H

#if defined(__cplusplus) || defined(__cplusplus__)
extern "C"
{
#endif

//-----------------------------------------------------------------------------
------------------
/**
 * Get TSMW IQ Interface version.
 *
 * @return int    32-Bit integer representing the 4-byte version code
Major.Minor.Patch.QFE
 */
int __cdecl TSMWGetVersion_c ( void );
```

```
//---------------------------------------------------------------------------
-----------------

/**

 * Get TSMW IQ Interface version as text

 *

 * @return char   Version as text

 */

char* __cdecl TSMWGetVersionText_c ( void );

//---------------------------------------------------------------------------
-----------------

/**

 * Initialize the TSMW IQ Data Interface DLL. Has to be called before any
other function is called.

 *

 * @return int    0 if successful

 */

int __cdecl TSMWInitInterface_c ( void );

//---------------------------------------------------------------------------
-----------------

/** Release the TSMW IQ Data Interface DLL. Shall be called before the dll
is unloaded.

 *

 * @return int    0 if successful

 */

int __cdecl TSMWReleaseInterface_c ( void );

//---------------------------------------------------------------------------
-----------------

/**

 * Get error message and error code.

 *

 * @param pErrorCode    Pointer to int where error code shall be stored,
zero if no error occured

 *

 * @return char*        Pointer to zero-terminated string with error
message.

 */

char* __cdecl TSMWGetLastError_c ( int *pErrorCode );

//---------------------------------------------------------------------------
-----------------

/**

 * Connect to the TSMW with the given ID.

 *
 * @param TSMW    0 if successful
 *
 * @return int
 */
/**
 * Connect to TSMW with given IP address
 *
 * @param IPAddress     IP address
 * @param pTSMWMode     TSMW mode structure
 * @param pTSMWID       Pointer to unsigned short which receives the TSMW ID
 *
 * @return int
 */
int __cdecl TSMWConnect_c (char* IPAddress, TSMW_IQIF_MODE_t *pTSMWMode,
unsigned short *pTSMWID);
//----------------------------------------------------------------------------
------------------
/**
 * Creates a new measurement session or addes/modifies filter specifications
for an existing measurement session.
 * Has to be called before any measurements is started.
 *
 * @param TSMWID           TSMW ID
 * @param pFilterSpec      Pointer to NSTR_TSMW_FILTER_SPEC structure
containing filter specification.
 *
 * @return int             0 if successful.
 */
int __cdecl TSMWIQSetup_c (unsigned short TSMWID,
                           TSMW_IQIF_FILTER_PARAM_t *pFilterParam,
                           long *plCoeff);
//----------------------------------------------------------------------------
------------------
/**
 * Start a new measurement.
 *
 * @param TSMWID           ID of TSMW to use for measurement
 * @param pMeasRequestID   Pointer to MeasRequestID variable, will receive
the measurement
 *                         request ID.
 * @param pStartTimes      Pointer to array with start times (optional), if
not used: NULL
 * @param NoOfStartTimes   Number of start times given
 * @param pMEAS_CTRL       Measurement control structure
 * @param pCHANNEL_CTRL1   channel control structure for RF channel 1
 * @param pCHANNEL_CTRL2   channel control structure for RF channel 2
 *
 * @return int             0 if successful.
 */
int __cdecl TSMWIQMeasure_c (unsigned short TSMWID, unsigned long
*pMeasRequestID,
                             unsigned __int64 *pStartTimes, long
NoOfStartTimes,
                             TSMW_IQIF_MEAS_CTRL_t *pMEAS_CTRL,
                             TSMW_IQIF_CH_CTRL_t *pCHANNEL_CTRL1,
                             TSMW_IQIF_CH_CTRL_t *pCHANNEL_CTRL2 );
//--------------------------------------------------------------------------
------------------
/**
 * Not yet implemented
 *
 * @param TSMWID
 * @param pMeasRequestID
 * @param pStartTimes
 * @param NoOfStartTimes
 * @param pStreamCtrl
 * @param pMEAS_CTRL
 * @param pCHANNEL_CTRL1
 * @param pCHANNEL_CTRL2
 *
 * @return int
 */
int __cdecl TSMWIQStream_c (unsigned short TSMWID, unsigned long
*pMeasRequestID,
                            unsigned long long *pStartTimes, long
NoOfStartTimes,
                            TSMW_IQIF_MEAS_CTRL *pMEAS_CTRL,
                            TSMW_IQIF_CH_CTRL *pCHANNEL_CTRL1,
                            TSMW_IQIF_CH_CTRL *pCHANNEL_CTRL2,
                            TSMW_IQIF_STREAM_CTRL_t *pStreamCtrl);
```

```
//----------------------------------------------------------------------------
------------------
/**
 * Check how many measurement results are available.
 *
 * @param pNoOfIQResults   Pointer to long which receives the number of IQ
measurement results
 *                         available
 *
 * @return int             0 if successful
 */
int __cdecl TSMWIQDataAvailable_c (long *pNoOfIQResults);
//----------------------------------------------------------------------------
------------------
/**
 * Get measurement result parameters.
 *
 * @param MeasRequestID    Measurement request ID. If 0, get parameters of
next available measurement
 *                         data block (with lowest measurement request ID)
 * @param TimeOut          Max. timeout to wait for measurement result
 * @param pIQResult        Pointer to structure to fill.
 *
 * @return int
 */
int TSMWIQGetResultParam_c (unsigned int MeasRequestID, unsigned int
TimeOut, TSMW_IQ_RESULT *pIQResult);
//----------------------------------------------------------------------------
------------------
/**
 * Get measurement result data.
 *
 * @param TSMWID           TSMW ID
 * @param MeasRequestID     0: Get measurement result with lowest
measurement request ID.
 *                         >0: Get result with specified measurement request
ID
 * @param TimeOut          Max. time in ms to wait for result
 * @param pIQResult        Pointer to TSMW_IQ_RESULT structure, will be
filled with result parameters,
 *                         see also TSMWIQGetResultParam_c
 * @param pReal            Pointer to short array to receive real part of
data
 * @param pImag            ... imag part
 * @param pScaling         Pointer to variable to receive scaling factor
 * @param pOvfl            ... overflow indicator
 * @param pCalibrated      ... "is calibrated" indicator
 * @param NoOfSamples      Number of samples reserved in sample buffer
 * @param NoOfChannels     Number of channels reserved in sample buffer
 * @param StartSampleDiv8  Shall be 0
 * @param NoOfUnpckBitRes  Shall be 0
 *
 * @return int
 */
int __cdecl TSMWIQGetDataInt16_c (unsigned short TSMWID, unsigned int
MeasRequestID,
                                  unsigned int TimeOut, TSMW_IQ_RESULT
*pIQResult,
                                  short* pReal, short* pImag, short*
pScaling, unsigned long* pOvfl,
                                  unsigned int *pCalibrated,
                                  unsigned int NoOfSamples, unsigned int
NoOfChannels,
                                  int StartSampleDiv8, int NoOfUnpckBitRes);
//--------------------------------------------------------------------------
------------------
// See TSMWIQGetDataInt16_c
int __cdecl TSMWIQGetDataInt32_c (unsigned short TSMWID, unsigned int
MeasRequestID,
                                  unsigned int TimeOut, TSMW_IQ_RESULT
*pIQResult,
                                  long* pReal, long* pImag, short* pScaling,
unsigned long* pOvfl,
                                  unsigned int *pCalibrated,
                                  unsigned int NoOfSamples, unsigned int
NoOfChannels,
                                  int StartSampleDiv8, int NoOfUnpckBitRes);
//--------------------------------------------------------------------------
------------------
// See TSMWIQGetDataInt16_c
int __cdecl TSMWIQGetDataSingle_c (unsigned short TSMWID, unsigned int
MeasRequestID,
                                   unsigned int TimeOut, TSMW_IQ_RESULT
*pIQResult,
                                   float* pReal, float* pImag, short*
pScaling, unsigned long* pOvfl,
                                   unsigned int *pCalibrated,
                                   unsigned int NoOfSamples, unsigned int
NoOfChannels,
                                   int StartSampleDiv8, int
NoOfUnpckBitRes);
//--------------------------------------------------------------------------
------------------
// See TSMWIQGetDataInt16_c
int __cdecl TSMWIQGetDataDouble_c (unsigned short TSMWID, unsigned int
MeasRequestID,
                                   unsigned int TimeOut, TSMW_IQ_RESULT
*pIQResult,
                                   double* pReal, double* pImag, short*
pScaling, unsigned long* pOvfl,
                                   unsigned int *pCalibrated,
                                   unsigned int NoOfSamples, unsigned int
NoOfChannels,
                                   int StartSampleDiv8, int
NoOfUnpckBitRes);
//--------------------------------------------------------------------------
------------------
// See TSMWIQGetDataInt16_c
int __cdecl TSMWIQGetDataSingleInterleaved_c (
   unsigned short TSMWID, unsigned int MeasRequestID,
   unsigned int TimeOut, TSMW_IQ_RESULT *pIQResult,
   float* pReal, short* pScaling, unsigned long* pOvfl,
   unsigned int *pCalibrated,
   unsigned int NoOfSamples, unsigned int NoOfChannels,
   int StartSampleDiv8, int NoOfUnpckBitRes);
//--------------------------------------------------------------------------
------------------
// See TSMWIQGetDataInt16_c
int __cdecl TSMWIQGetDataDoubleInterleaved_c (
   unsigned short TSMWID, unsigned int MeasRequestID,
   unsigned int TimeOut, TSMW_IQ_RESULT *pIQResult,
   double* pReal, short* pScaling, unsigned long* pOvfl,
   unsigned int *pCalibrated,
   unsigned int NoOfSamples, unsigned int NoOfChannels,
   int StartSampleDiv8, int NoOfUnpckBitRes);
//--------------------------------------------------------------------------
------------------
```

```
// Shutdown TSMW
int __cdecl TSMWShutdown_c (unsigned short TSMWID);
//---------------------------------------------------------------------------
------------------
// Enable tracking generator
int __cdecl TSMWTGEnable_c (unsigned short TSMWID, unsigned int Frontends);
//---------------------------------------------------------------------------
------------------
// Set proc.board DAC for ref.clock
int __cdecl TSMWSetDAC_c (unsigned short TSMWID, short Value);
//---------------------------------------------------------------------------
------------------
// Enable/disable GPS synchronization
int __cdecl TSMWGPSSync_c (unsigned short TSMWID, int Enable);
//---------------------------------------------------------------------------
------------------
// Enable/disable GPS data subscription from TSMW
int __cdecl TSMWGPSEnable_c (unsigned short TSMWID, int NoOfBufferedLines,
int Enable);
//---------------------------------------------------------------------------
------------------
// Clear GPS NMEA buffer
int __cdecl TSMWGPSClearBuffer_c ( void );
//---------------------------------------------------------------------------
------------------
// Get NMEA data line
char* __cdecl TSMWGPSGetNMEALine_c ();
//---------------------------------------------------------------------------
------------------
// Get IQ time of given TSMW
int __cdecl TSMWGetIQTime_c (unsigned short TSMWID, unsigned long long
*pIQTime);
//---------------------------------------------------------------------------
------------------
// Set/read TSMW trigger IOs
int __cdecl TSMWTrigger_c (unsigned short TSMWID, int TriggerLine, int Out,
                           int *pTrig1Value, int *pTrig2Value);

#if defined(__cplusplus) || defined(__cplusplus__)
}
#endif
```

```
#endif // _TSMWIQINTERFACEFUNC_H
```

Index