Faculty of Engineering LTH

Digital Projects
Lund University, Faculty of Engineering

Lund 2007-05-19

SMS-Relay

Supervisor:
Bertil Lindvall

Authers:
Per Wallmark, e03pw@student.lth.se
Gustaf Persson, e02gp@student.lth.se

Abstract

The aim of this project is to construct a fully functional SMS-relay device. This device enables you to remotely switch on or off any of your home electronics or appliances such as refrigerators, heaters or computers. An additional feature of this device is that it can report back a “status report” to the user, the communication is done through standard SMS on any cellular phone. It’s of course also possible so switch the output by simply pressing the corresponding button.

The project has been successful and finished product is even better than we expected. If this sounds interesting please study our reference design (attached schematic) and the source-code.

Innehållsförteckning

1 Projektbeskrivning

Målet med projektet är att skapa ett SMS-styrt relä, detta ska kunna användas för att t.ex. styra värme och belysning. Mer ingående, genom att skicka ett SMS styra fyra oberoende utgångar.

1.1 Kravspecifikation

Fasta krav:

Den färdiga produkten ska kunna ta emot ett SMS och beroende på innehållet styra flertalet utgångar.

I mån av tid:

Då de fasta kraven är uppfyllda kan systemet utökas med, ej nödvändigtvis i denna ordning, följande funktioner.

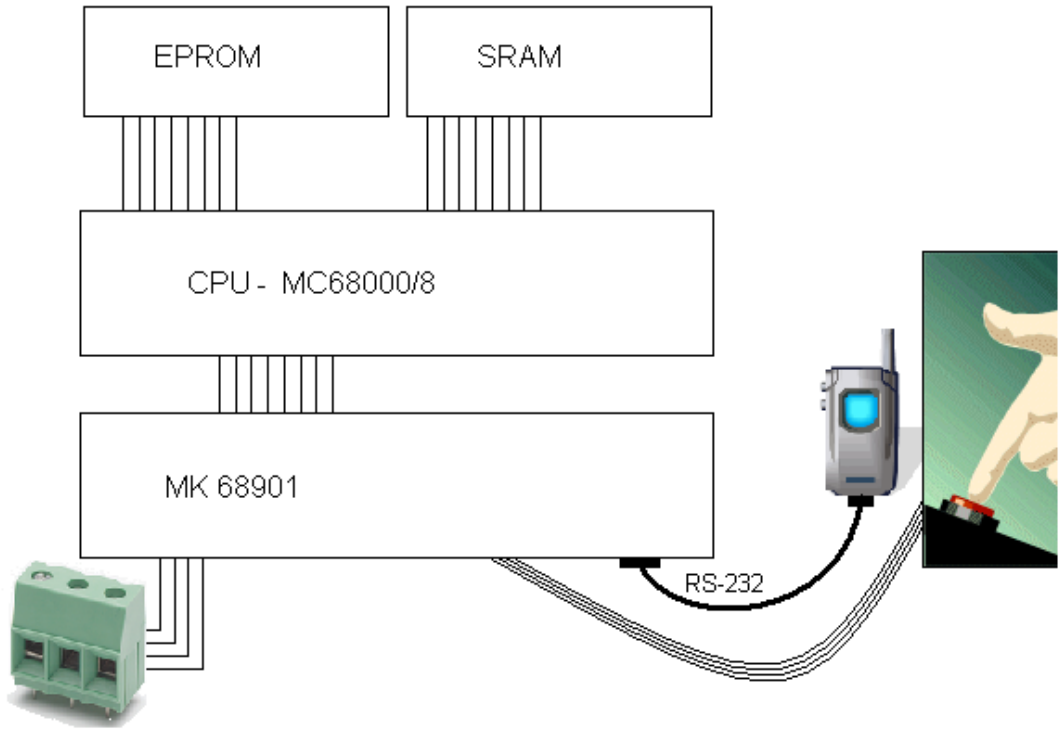
- Systemet svarar med ett SMS med nuvarande status om det är önskat i det mottagna meddelandet.
- Temperatur inom/utomhus kan avläsas och läggas till i nuvarande status.
- En manuell möjlighet att styra utgångarna.
- Lysdioder för status på utgångar.

2 Beskrivning av lösningen

Systemet är byggt kring en Motorola 68008, ett EPROM och SRAM på 64kb. Till det har vi för kommunikationen med telefonen en multifunktions krets MK-901, den omvandlar den parallella databitarna från CPUn till seriella, eftersom telefonen vill ha informationen så. Den hanterar också avbrotten från de manuella knapparna. Kommunikationen mot telefonen sker över en seriekabel, där vi skickar så kallade AT-kommandon som delvis är generella för alla typer av modem.
Telefonen initieras för att kunna vidarebefordra ett inkommande SMS direkt till seriekabeln, ett avbrott genereras för varje överförd symbol och vi lagrar detta i en buffer. Vi kontrollerar kontinuerligt ifall ett radbrytstecken har tagits emot. Ifall ett radbryt upptäcks kontrolleras den mottagna strängen/kommandot för att se om någon åtgärd ska vidtagas eller inte. Detta är en kortfattad beskrivning av hur vår mjukvara fungerar. Mer utförlig beskrivning kan läsas ut av flödesschemat nedan (figur 2 och 3).

2.1 Hårdvara

2.1.1 Blockschema



Figur 1 Blockschema - hårdvara

2.1.2 Komponentval

2.1.2.1 CPU

Vi använde oss av en av standardprocessorerna i kursen Motorola 68008. Den har 8-bitars databuss och kan adressera upp till en Mb. Maximal klockfrekvens är 10 MHz. Den har tre avbrottsnivåer 7, 5 och 2 där de högre även används internt. Vi använder vektoriserade avbrott på nivå 2.

2.1.2.2 Minne

Som EPROM använde vi 27C64 som rymmer 64 kb vilket räcker till vår applikation. Till SRAM använde vi en 6226 på 64kb.

2.1.2.3 Multifunktion IO

Multifunktionskretsen MK-901 använd för parallell- till seriell- omvandling, genererar de vektoriserade avbrotten till CPU'n, både för knapparna och de mottagna databitarna. Den innehåller även timers som används för att ställa in baudrate för seriell kommunikation. Den har 8 I/O ben som kan konfigureras valfritt, vi använder 4 för de fyra knapparna och fyra är kopplade till reläerna som vi vill styra.

2.1.2.4 Drivkrets

Vi använder en ULN280x Eight Darlington array för att driva både lysdioder och reläerna. Detta eftersom det visade sig bli lite för stor last att dra från Multifunktionskretsens utgångar. Kretsen används normalt för att driva motorer.

2.1.2.5 Logikkretsar

PALCE22V10 Programmable Array Logic används som Bus Controller, både till adressbuss och databuss. Den används även för att generera styrsignaler som processorn behöver, exempelvis IACK, DS, DTACK mm.

En 74HC04 hex inverter används i uppkopplingen vid kristalloscillatorn för att generera klockpulsen. Vi använder även denna för att invertera utgångarna från multifunktionskretsen, detta för att styra den röda färgen i RGB-lysdioderna.

2.1.2.6 Reset-krets

Vår applikation kräver pålitlighet och bör därför klara av ett strömavbrott. Resetkretsens syfte är att generera en fördröjd reset-signal som ska starta om processorn när spänningen åter har stabiliserats. Vi använder till detta en TL7705.

2.1.2.7 Spänningsregulator

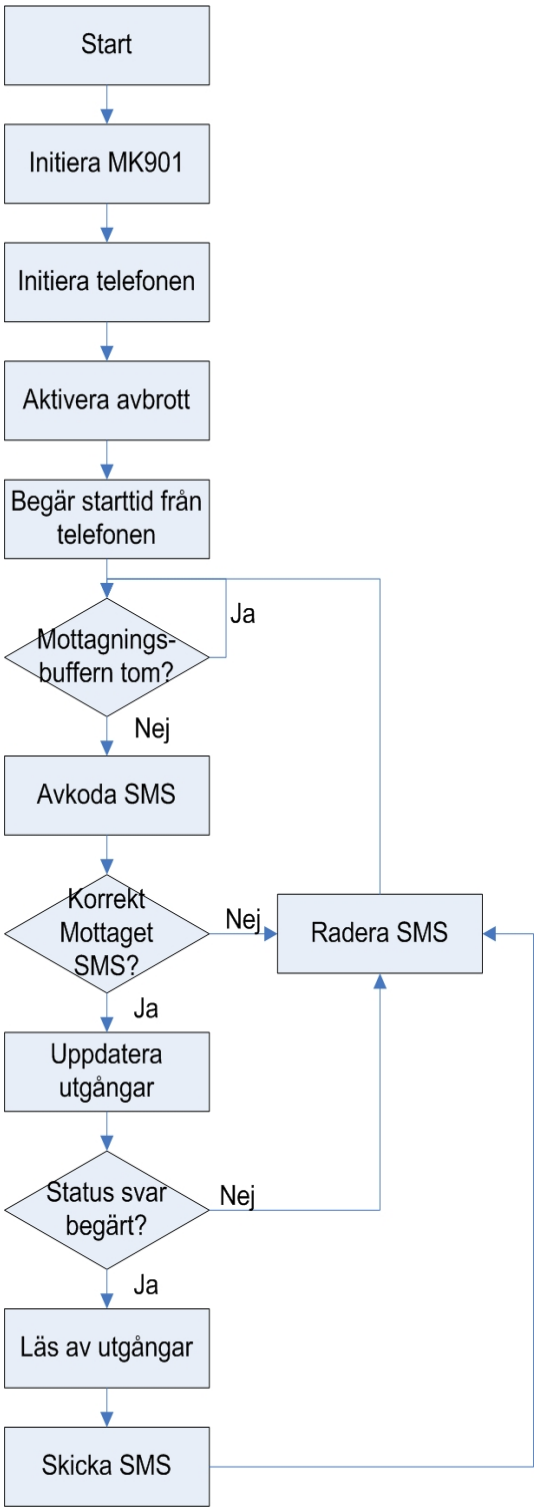
För att generera en stabil +5 volts signal använde vi först en LM7805. Då den blev väldigt varm, även med kylfläns monterad, bestämde vi oss för att istället använda en switchad spänningsregulator. Till detta används en LM2574.

2.1.2.8 Rs-232

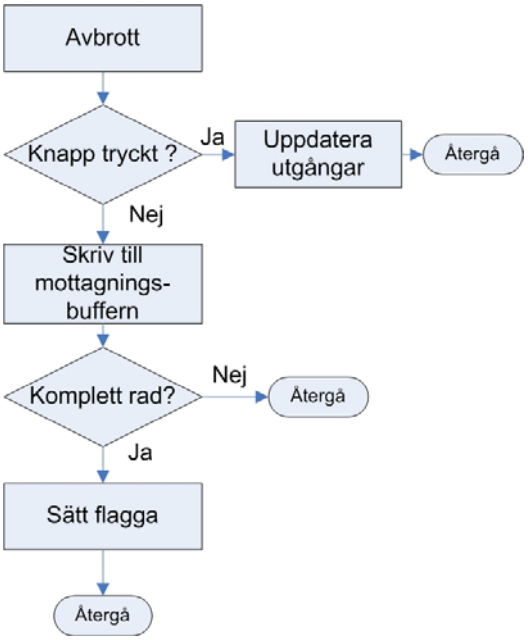
En RS-232 krets behövs för att konvertera TTL -nivåerna som används av processorn till CMOS som telefonen använder.

2.2 Mjukvara

2.2.1 Flödesschema



Figur 2 Flödesschema för huvudprogrammet



Figur 3 Flödesschema för avbrott

2.3 Problem

2.3.1 Hårdvara

Efter att vi kopplat och virat färdigt vår grundkonstruktion var det dags för testning. Till vår och Bertils stora förvåning hade vi lyckats få alla virningar rätt vilket var en smula ovanligt. Våra problem började vid ett senare skede då vi försökte få liv i seriell-kommunikationen. Till vår stora besvikelse så kunde inte den sofistikerade multifunktionskretsen själv generera en lämplig baudrate. Vi var därför tvungna att byta ut vår huvudkristall från 10Mhz till en frekvens som var jämt delbar med vår valda baudrate på 9600bps. Kristallen i fråga har frekvensen 4,91520Mhz.

Vidare hade vi ytterligare svårigheter med seriekommunikationen då vi endast använda pinnarna: signal ground, Rx och Tx. Det visade sig att telefonen krävde en typ av hårvaruhandskakning. Detta löste vi genom att bygla mellan 2 par ben som resulterar i att en request från telefonen direkt blir ett klartecken tillbaka till telefonen (detaljerna syns på kopplingsschemat).

När vi senare skulle frångå emulatorkopplingen och skulle börja köra med egen processor så visade det sig att vi trots allt hade vissa tvivelaktiga virningar. Vi hade bland annat kopplat samman två TTL-nivå-signaler till en ingång. Tydligen hade emulatorkopplingen en in-inpedans som kompenserat för denna felkoppling. När vår krets skulle klara sig på egen hand gick det däremot inte så bra. Detta löstes snabbt då vi insett vårt misstag, genom att koppla båda signalerna till vår logik där vi kör en eller-funktion på dessa.

Nästa problem moment var när vi ville addera spänningsregulatorn för att driva vår konstruktion. Vi valde en standardkrets, LM7805, den visade sig utveckla på tok för mycket värme, även med kylfläns och fick bytas ut mot en switchad spänningsregulator.

2.3.2 Mjukvara

Ingen av gruppmedlemmarna hade tidigare erfarenhet från c-programmering, så inledningsfasen var lite trevande. Vi körde enligt ”trial-and-error” principen utgående ifrån det programskelett som vi fått. Trots vår okunskap i c-programmering så flöt arbetet på förvånansvärt bra.

Det svåraste momentet var att läsa ut och ”översätta” själva texten ifrån ett inkommet SMS. När väl detta var löst så applicerades metoden baklänges för att kunna generera våra svars-SMS. För att sedan få iväg det SMS som vi ”kompilerat” krävdes det noggrann bit-räkning eftersom telefonen kräver att i förväg få reda på hur stort meddelande som ska sändas. SMS innehållet är kodat med så kallad PDU-kod, vi har importerat en funktion som vi hittade på Internet som sköter den delen av kodningen, men på grund av begränsade resurser i kompilator och processorn var vi tvungna att skriva om stora delar av funktionen.

3 Slutsats / resultat

Vi blev klara ganska tidigt med grundkonstruktionen, därefter har mycket tid lagts till att finslipa både hårdvaran och mjukvaran. Eftersom vi tänkt använda konstruktionen ”på riktigt” så har mycket tid och möda lagts ner för att få den att fungera problemfritt och pålitligt. Slutresultatet är vi mycket nöjda både funktionellt men även formgivningsmässigt, eftersom vi byggt in konstruktionen i en passande svart låda.

Vi har dock en kvarstående ”bug” som vi inte riktigt lyckats sätta fingret på vad orsaken är. Problemet består i att vissa, förefallande slumpmässigt utvalda, SMS totalt ignoreras av vår konstruktion. Det resulterar inte i att systemet hänger sig eller annat oönskat beteende utan allt fortskrider som vanligt. Det gör ju att det är svårt att hitta exakt vad det är som går snett. Våra misstankar har delvis varit riktade mot telefonen men det är nog mer troligt att det är något specialfall som vi inte tänkt på.

Vi är överlag mycket positiva till kursen som har varit mycket givande. Att få göra något ”riktigt” till skillnad från alla teoretiska kurser man läst här på LTH. Kursen kan varmt rekommenderas, se till att läsa den innan du lämnar skolan on du inte redan har gjort det!

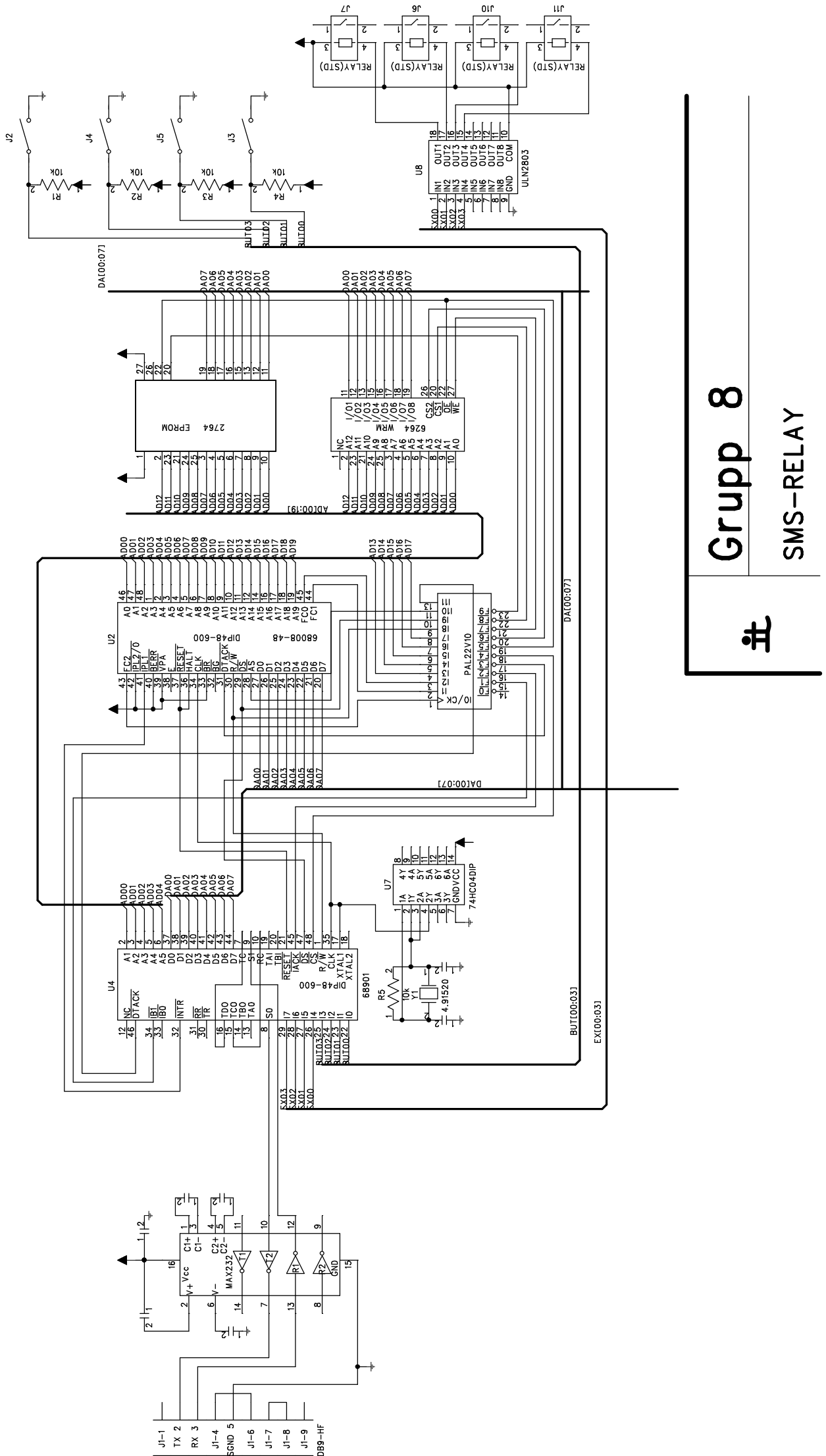
4 Referenser

Datablad:
http://www.it.lth.se/digp/datablad/datablad_index.asp
http://pupius.co.uk/download/misc/t68i-at-commands.pdf

Källkod:
http://www.nerdlabs.org/downloads/pduconv/pduconv-0.1.zip

5 Bilagor

Bilaga 1 – Kopplingsschema från PowerLogic.


Grupp 8
SMS-RELAY
RELAY(STD)
ULN2803
2764 EPROM
6264 WRM
68008-48
DIP48-600
PAL22V10
74HC04DIP
68901
4.91520
MAX232
DB9-HF
DAI00:07]
DAI00:07]
BUTI00:03]
EXI00:03]
ADI00:19]